University of Washington

Abstract

Optimization of Conceptual Aircraft Design for
Stability/Control and Performance

John Douglas Galloway, Jr.

Chairperson of the Supervisory Committee: Professor Eli Livne
Department of Aeronautics and Astronautics

The goal of this research was to develop a design optimization capability for the conceptual design of airplanes, where static and dynamic stability and control requirements are addressed simultaneously together with mission and performance requirements. Typically, these requirements are not even considered until after initial aircraft sizing is completed. A design program was created to size the airplane, determine dimensions and locations of tail surfaces (horizontal and vertical tails), control surfaces (elevator, rudder, ailerons), and landing gear, and distribute systems and components all in one step in an optimal manner. Analysis methods used in the calculation of applied aerodynamics, weight and balance, mission analysis, performance, and static / dynamic stability and control are all computationally fast, leading to an analysis module that is extremely efficient for optimization or parametric studies. Integration of the optimization modules with the analysis module is done in a way that makes it possible to easily change selections of design variables, constraints, and objective function for different design optimization studies. Results of the analysis program were compared with data from existing aircraft to verify accuracy. These results and those from a number of optimization test cases are included in this thesis. In its present form the new capability is a valuable addition to the computer design tools available for conceptual design of general aviation airplanes. It will be very useful for educational purposes, especially, in undergraduate and graduate airplane design courses.

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	12.Feb.01	THESIS

4. TITLE AND SUBTITLE
OPTIMIZATION OF CONCEPTUAL AIRCRAFT DESIGN FOR STABILITY/CONTROL AND PERFORMANCE

5. FUNDING NUMBERS

6. AUTHOR(S)
2D LT GALLOWAY JOHN D

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
UNIVERSITY OF WASHINGTON

8. PERFORMING ORGANIZATION REPORT NUMBER
CI01-47

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
THE DEPARTMENT OF THE AIR FORCE
AFIT/CIA, BLDG 125
2950 P STREET
WPAFB OH 45433

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Unlimited distribution
In Accordance With AFI 35-205/AFIT Sup 1

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

14. SUBJECT TERMS

15. NUMBER OF PAGES
198

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

Optimization of Conceptual Aircraft Design for Stability/Control and Performance

John Douglas Galloway, Jr.

A thesis submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Aeronautics and Astronautics

University of Washington

2000

Program Authorized to Offer Degree: Department of Aeronautics and Astronautics

University of Washington
Graduate School

This is to certify that I have examined this copy of a master's thesis by

John Douglas Galloway, Jr.

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

Eli Livne

Uy Loi Ly

Date: 8/9/2010

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of this thesis is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Any other reproduction for any purposes or by any means shall not be allowed without my written permission.

Signature John D. Fallon Jr

Date 14 Aug '00

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE LIST

AR	wing aspect ratio
b	wing span
b_a	aileron span
b_e	elevator span
b_{flap}	flap span
b_r	rudder span
$BSFC$	brake specific fuel consumption
b_t	horizontal tail span
b_v	vertical tail span
c_a	aileron chord
C_{D_0}	parasite drag coefficient
c_e	elevator chord
C_F	aerodynamic cleanness coefficient
c_{flap}	flap chord
$C_{L_{max}}$	maximum lift coefficient
c_r	rudder chord
c_{root}	wing root chord
c_{rtail}	horizontal tail root chord

c_{rvt}	vertical tail root chord
D_{prop}	propeller diameter
e	Oswald's efficiency factor
g	acceleration due to gravity
g_{effect}	ground effect factor
h	maximum fuselage height
h_{bcy}	fuselage height at the back of the canopy
h_{fcy}	fuselage height at the front of the canopy
h_{nose}	fuselage height at the nose region
$h_{obstacle}$	obstacle height at takeoff or landing
H_t/H_v	0.0 for conventional tail; 1.0 for "T" tail
i_{wing}	wing incidence angle
l	rolling moment
L_B, l_b	fuselage length
$(L/D)_{max}$	maximum lift-to-drag ratio
L_m, l_m	main gear length
l_{mh}	distance from the nose to the point of max fuselage height
L_n, l_n	nose gear length
l_t	distance from the center of gravity to the quarter chord of the horizontal tail
L_t, l'_t, l_{t_1}	distance from wing quarter-MAC to tail quarter-MAC

M	Mach number
m	pitching moment
n	yawing moment
N_{en}	number of engines
N_l	ultimate landing load factor
N_p	number of personnel onboard (crew and passengers)
N_t	number of fuel tanks
N_z	ultimate load factor; = 1.5 * limit load factor
p	roll rate
P_{delta}	cabin pressure differential
P_{max}	maximum power at sea level
q	pitch rate
q_D	dynamic pressure
r	yaw rate
r_1	fuselage radius in the vicinity of the vertical tail
Range	distance of a specified cruise leg
RaymerA	coefficient used in empty weight approximation, refer to RA92
RaymerC	coefficient used in empty weight approximation, refer to RA92
RoskamA	coefficient used in empty weight approximation, refer to RO285
RoskamB	coefficient used in empty weight approximation, refer to RO285
R_x	x-axis non-dimensional radii of gyration

R_y	y-axis non-dimensional radii of gyration
R_z	z-axis non-dimensional radii of gyration
S_f	fuselage wetted area
S_{ht}	horizontal tail area
S_{vt}	vertical tail area
S_w, S	wing area
t/c	thickness-to-chord ratio of the wing
throttle	throttle setting
U	airspeed, subsript *stall* indicates stall airspeed
V_i	integral fuel tanks volume
V_{pr}	volume of pressurized section of the fuselage
V_t	total fuel volume
W	aircraft weight within a particular mission leg
w_{bcy}	fuselage width at the back of the canopy
W_{crew}	total weight of the crew
W_{dg}	aircraft design gross weight
W_{en}	weight per engine
w_{fcy}	fuselage width at the front of the canopy
W_{final}	final weight after completing mission analysis
W_{fus}	fuselage width
w_{fus}	maximum fuselage width

W_{fw}	weight of fuel in wing
W_{guess}	initial guess of aircraft weight for sizing
W_l	landing design gross weight
$W_{passengers}$	total weight of the passengers
$W_{payload}$	payload weight
W_{total}	total aircraft, = sum of component weights
W_{uav}	uninstalled avionics weight
x'	distance from the center of gravity to the wing quarter chord (+ for center of gravity ahead of the wing quarter chord)
x_a	distance from the center of gravity to the wing quarter chord (+ for center of gravity behind wing quarter chord)
Y_i	distance from the body centerline to the inboard edge of the ailerons
z_j	perpendicular distance between the center of gravity and the thrust line (+ for the center of gravity above the thrust line)
z_v	distance from the center of pressure of the vertical tail to the airplane's x-axis (+ for the vertical tail above the x-axis)
z_w	distance from body centerline to quarter-chord point of the exposed wing root chord (+ for the quarter-chord point below the body centerline)
$\Delta C_{D_{flap}} / \Delta C_{D_0}$	ratio to account for the increase in drag coefficient due to flaps
$\Delta C_{D_{gea}} / \Delta C_{D_0}$	ratio to account for the increase in drag coefficient due to landing gear
Δh	change in altitude during climb leg
Δn	parameter used in calculating landing distance, refer to TO82
$\bar{\gamma}$	parameter used in calculating landing distance, refer to TO82

Γ	wing dihedral angle
Λ	wing sweep angle
Λ_h	horizontal tail sweep angle
Λ_{vt}	vertical tail sweep angle
δ_A	aileron deflection, positive aileron deflection is defined as that which causes a positive rolling moment (right stick)
δ_E	elevator deflection, positive elevator deflection is defined as that which causes a positive pitching moment (aft stick)
δ_R	rudder deflection, positive rudder deflection is defined as that which causes a positive yawing moment (right rudder pedal)
η_p	propeller efficiency
η_t	efficiency of the horizontal tail
ρ	air density, subscript *SL* indicates sea level conditions

ACKNOWLEDGEMENTS

A great amount of gratitude and appreciation is extended to my advisor, Professor Eli Livne for his great amount of knowledge, vision, and understanding. A sincere "Thank you," goes to him for first agreeing to allow me to work for him in pursuing this program and this research in the allotted ten-and-a-half months, and secondly for seeing me through it. The same gratitude is extended to the Department of Aeronautics and Astronautics at the University of Washington and the Department of Aeronautical Engineering at the Air Force Academy for their sponsorship and faith.

Any substantial challenge does not come without sacrifices. For this reason, I lovingly thank my wonderful wife, Dene', who undoubtedly sacrificed the most over the last year while providing her utmost support.

1. INTRODUCTION

Accounting for stability and control requirements in the course of conceptual design of airplanes is usually done after the initial sizing of the airplane (for wing loading, thrust or power loading and take-off gross weight) is completed (RA92, RO185, TO82). When stability and control constraints are first considered, particularly in an educational context in most undergraduate airplane design courses, only static stability and control requirements are usually addressed first, and control surfaces are frequently sized by using statistical equations based on "good practices" of the past (RA92).

Dynamic stability and control requirements are usually not addressed at all early in the design process. They are usually examined when the overall configuration is almost frozen, based on the common assumption that it is possible to solve any problems in the area of dynamic stability and control by minor modifications of the configuration or by implementation of some forms of active control.

The challenge of designing an active control system to correct deficiencies of a given airplane configuration is then delegated to controls engineers who might find it extremely difficult to design an efficient control system for the configuration they have. If only it were possible to design the configuration of the airplane and its control system simultaneously, so that they affect each other, the resulting configuration will evolve to be more control-friendly, the control system more efficient, and the combined product more efficient.

Advantages of integrated airplane configuration / control system design optimization in conceptual design have been recognized for quite a while (AM60, MK90, MO92). Even in the case of airplanes with no active controls, the utilization of optimization techniques early in the design process to search for designs that meet a large set of design requirements (including dynamic stability and control constraints) can lead to desirable designs and prevent subsequent costly modifications.

The goal of the work described here is to develop a design optimization capability for the conceptual design of airplanes, where static and dynamic stability and control requirements are addressed simultaneously together with mission and performance requirements. The desire is to size the airplane, determine dimensions and locations of tail surfaces (horizontal and vertical tails), control surfaces (elevator, rudder, ailerons), and landing gear, and distribute systems and components all in one step in an optimal manner. Analysis methods used in the areas of applied aerodynamics, weight and balance, mission analysis, performance, and static / dynamic stability and control are all computationally fast, leading to an analysis module that is extremely efficient for optimization or parametric studies. Integration of the optimization modules with the analysis module is done in a way that makes it possible to easily change selections of design variables, constraints, and objective function for different design optimization studies.

The present capability constitutes a first step on the road to the development of integrated airframe / control system optimization for the conceptual design phase. It is, thus, limited

in a number of ways. First, as already stated above, active control design is not part of it yet. In addition, at this stage, modeling is limited to general aviation airplanes, and thrust effects are not included in the dynamic and static stability and control evaluations. The computer capability developed requires the usage of a particular system of engineering units (the English system, following SM84). All these limitations can be removed easily at a later stage.

The thesis opens with a sequence of chapters describing parameterization of the airplane for design purposes and all aspects of the design-oriented analysis. Chapter 2 discusses design parameterization and the selection of pre-assigned parameters and design variables. Chapter 3 describes weight and balance estimation by component build up. Chapter 4 focuses on aerodynamics for mission performance, and Chapter 5 describes mission analysis and performance evaluation. Chapter 6 discusses creation of important geometrical entities required in stability derivatives calculation. It is followed by Chapter 7, which lists all stability derivatives used, and the equations used to evaluate them. Chapters 8 and 9 discuss static and dynamic stability and control behavior, respectively. Discussion of the optimization problem formulation, integration of analysis and optimization modules, and a list of all behavior and geometric constraints available in the new capability is provided in Chapter 10. Chapter 11 includes analysis results from the analysis part of the new capability, demonstrating its accuracy and reliability when applied to real airplane configurations. Chapter 12 presents design test cases and optimization results obtained with the new capability and a discussion of the importance

of integration of static and dynamic stability and control requirements into the conceptual design optimization of airplanes. Chapter 13 concludes the thesis with a summary of major achievements and insights gained, and recommendations for further developments. A comprehensive bibliography contains all sources of information used for this work.

2. AIRCRAFT DESIGN PARAMETERIZATION

The set of parameters defining any given design can be divided into pre-assigned parameters (PPs), those parameters which are pre-determined and do not change during the design, and design variables (DVs), those parameters that are varied by the optimization process. The following chapter describes the design variables and the pre-assigned parameters used to define an airplane in this work.

2.1 Design Variables

The airplane configuration design is defined by thirty design variables, which are listed in Table 2.1.1. These variables cover the aircraft geometric features (including wing, fuselage, tail surfaces, control surfaces, landing gear and location of major systems), engine, and weight. The notation used in this table is consistent with the actual *MATLAB* variables and coincides with the layout of the actual input file. Refer to Figure 2.1 for a graphical definition of each of the geometric design variables.

TABLE 2.1.1: Design Variables

NAME	DESCRIPTION
B	Wing span (ft)
CROOT	Wing root chord (ft)
TR	Wing taper ratio
DIH	Dihedral angle (deg) (+ for dihedral, - for anhedral)
Xlewing	Distance of the l.e. of the wing from the nose (ft)
BT	Horizontal tail span (ft)
Crtail	Horizontal tail root chord (ft)
TRT	Horizontal tail taper ratio
Xletail	Distance of the l.e. of the horiz. tail from the nose (ft)
BV	Vertical tail span (ft)
Crvt	Vertical tail root chord (ft)
TRvt	Vertical tail taper ratio
Xlevt	Distance of the l.e. of the vert. tail from the nose (ft)
BA	Aileron span (ft)
CA	Aileron chord (ft)
YI	Distance from the body centerline to the inboard edge of the aileron
BFlap	Flap span (ft)
CFlap	Flap chord (ft)
BE	Elevator span (ft)
CHE	Elevator chord (ft)
BR	Rudder span (ft)
CR	Rudder chord (ft)
LB	Fuselage length (ft)
Lm	Length of main landing gear (in)
CGmgr	x-position of the main landing gear c.g. (ft)
Ln	Length of nose gear (in)
CGngr	x-position of the nose gear c.g. (ft)
Pmax	Maximum power at sea level (hp)
Wguess	Initial guess of aircraft weight for sizing (lb)
XM	Distance of c.g from the nose (ft)

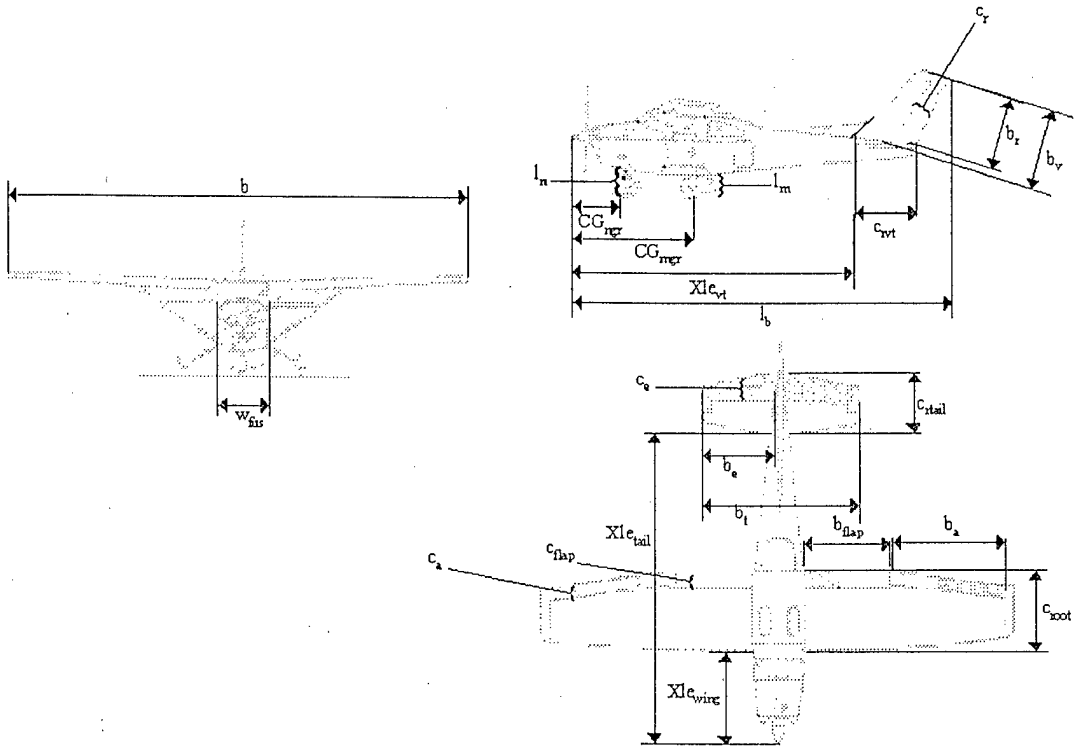


Figure 2.1. Definition of Design Variables

2.2 Preassigned Parameters

Apart from the design variables, there are eighty-eight permanent parameters that must be defined. The list of these parameters may be found in Table 2.2.1 under the same notation used for the actual *MATLAB* variables. These pre-assigned parameters cover five major areas of interest including overall aircraft geometry / configuration, aerodynamics, power, crew and payload, and weight estimation. The pre-assigned

parameters are defined once at the beginning of an optimization run and are then held fixed throughout the optimization process.

TABLE 2.2.1: PREASSIGNED PARAMETERS

NAME	DESCRIPTION
Lo	Fuselage length (ft)
Ho	Max body height at wing-body intersection (ft)
H1o	Fuselage height 1/4 the fuselage length from nose (ft)
H2o	Fuselage height 3/4 the fuselage length from nose (ft)
HBCYo	Fuselage height at back of canopy (ft)
HFCYo	Fuselage height at front of canopy (ft)
Hgearo	Fuselage height at location of main landing gear (ft)
HNOSEo	Fuselage height at the nose region (ft)
LBCYo	Length from nose to back of canopy (ft)
LFCYo	Length from nose to front of canopy (ft)
LMHo	Length from nose to point of max fuselage height (ft)
R1o	Radius of fuselage in vicinity of vertical tail (ft)
WFUSo	Maximum fuselage width (ft)
WBCYo	Fuselage width at back of canopy (ft)
WFCYo	Fuselage width at front of canopy (ft)
WNOSEo	Fuselage width in the nose region (ft)
CGaco	x-Position of the air conditioning and anti-ice system c.g. (ft)
CGaviono	x-Position of the avioncis system c.g. (ft)
CGctrlo	x-Position of the flight control system c.g. (ft)
CGeleco	x-Position of the electrical system c.g. (ft)
CGengo	x-Position of the engine c.g. (ft)
CGfuelo	x-Position of the fuel system c.g. (ft)
CGfurno	x-Position of the furnishings c.g. (ft)
CGhydro	x-Position of the hydraulics system c.g. (ft)
slender	Fuselage slenderness ratio
Option	=1 uses given values;=2 scales to length; =3 scales to width
SA	Sweep angle (deg) (+ for sweep aft)
IWING	Incidence angle of the wing at the root chord (deg) (+ for upward incidence)
toverc	Airfoil thickness to chord ratio
SAH	Horizontal tail sweep angle (deg) (+ for sweep aft)

TABLE 2.2.1: PREASSIGNED PARAMETERS (continued)

ITAIL	Incidence angle of the horizontal tail at the root chord (deg) (+ for upward incidence)
SAvt	Sweep angle of the vertical tail at 25% MAC (deg) (+ for sweep aft)
engineDref	Reference engine diameter (in)
engineLref	Reference engine length (in)
CD0	Parasite drag
CF	Aero cleaness coefficient (calculated using Swet if = 0)
CDA2DW	2D drag curve slope of wing (1/rad)
cDFlapRatio	Drag accounting for flaps at takeoff
CDLGratio	cD adjustment for landing gear
cFriction	Skin friction drag coefficient
CLA2DT	2D lift curve slope of the horiz. tail (1/rad)
CLA2DW	2D lift curve slope of the wing (1/rad)
CLAFUS	Lift curve slope of fuselage (1/rad)
cLmax	Maximum lift coefficient
cLmaxClean	Maximum clean lift coefficient
EnginDragFactor	Drag accounting for engine installation
EFF	Horizontal tail efficiency factor
ETAV	Vertical tail efficiency factor
groundeffect	Ground effect factor
LDmax	Maximum lift-to-drag ratio
Oswald	Oswald's coefficient
AnPower	Dependency of power on altitude
enginePowerref	Reference engine power (hp)
Nen	Number of engines
Wen	Weight of each engine (lb)
Dprop	Propeller Diameter (inches)
Npassengers	Number of passengers
Ncrew	Number of crew members
Wppass	Weight per passenger w/ baggage (lb)
Wpcrew	Weight per crewman w/ equipment (lb)
Wpayload	Weight of payload (lb)
Growth	Growth factor for larger payloads
Wuav	Uninstalled avionics weight (typically = 800-1400 lbs)
RoskamA	Roskam coefficient for Wempty
RoskamB	Roskam coefficient for Wempty
RaymerA	Raymer coefficient for Wempty

TABLE 2.2.1: PREASSIGNED PARAMETERS (continued)

RaymerC	Raymer coefficient for Wempty
HtHv	0.0 for conventional tail & 1.0 for "T" tail
Nl	Ultimate landing load factor; = Ngear*1.5
Nstrut	Number of main landing gear struts
Fueldens	Fuel density (lb/gal)
Nt	Number of fuel tanks
Nz	Ultimate load factor; = 1.5 * limit load factor
Pdelta	Cabin pressure differential; typically 8 psi (lb/in^2)
Vpr	Volume of pressurized section (ft^3)
Wl	Landing design gross weight (lb)
eps	Tolerance for weight mismatch
ZA	Vertical distance from the c.g. to the wing a.c. (ft)
ZT	Perpendicular distance from the c.g. to the thrust line (ft)
ZV	Distance from the c.p. of the vertical tail to the airplane's x-axis (ft)
ZW	Distance from body centerline to quarter-chord point of exposed wing root chord (ft)
ALPHA	Angle of attack (deg)
GAMMA	Flight path angle (deg)
COSXZ	Cosine of the angle between the thrust axis and the wind axis
SINXZ	Sine of the angle between the thrust axis and the wind axis
UpperMach	Upper Mach limit
WindMax	Max wind speed at altitude (ft/s)
GRAVIT	Gravitational acceleration (ft/s^2)

Not all design variables have to be used in a particular run. The capability described here allows for assigning a constant value for any design variable by adding it to the list of pre-assigned parameters.

2.3 Geometric Calculations

As can be noticed in an examination of Tables 2.1.1 and 2.2.1, two important parameters (area and aspect ratio), which are generally used to specify geometry characteristics of airplanes, are not included in the lists. Areas of wing and control surfaces are determined by the respective root chord, c_r, taper ratio, λ, and span, b. The corresponding aspect ratios are determined by the span, b, and projected area. Once values are assigned to a complete set of DVs and PPs in the course of optimization, the wing area and aspect ratio are then calculated from

$$S = \frac{c_r(1+\lambda)}{2} b \tag{2-1}$$

The control surfaces are assumed to be rectangular in the present capability. Control surface areas are simply the span multiplied by the chord. Aspect ratio of the lifting surfaces is given by:

$$AR = \frac{b^2}{S} \tag{2-2}$$

3. AIRCRAFT WEIGHT ANALYSIS

The weight analysis section estimates the total gross weight, the center of gravity location, and the moments of inertia of the aircraft. The methods used are based on statistical weight equations for components (given here for general-aviation aircraft, RA92,RO285).

3.1 Fuel Weight in Wing Tanks

An important feature of any design is the ability to carry the necessary amount of fuel. The volume available from wing integral fuel tanks is determined by (TO82):

$$V_i = 0.54 \frac{S^2}{b} \left(t/c\right) \cdot \left(\frac{1+\lambda+\lambda^2}{(1+\lambda)^2} \right) \tag{3-1}$$

The fuel weight is then found by multiplying the result by the appropriate fuel density.

3.2 Component Weight Breakdown

The component weight breakdown estimates the total weight using statistical equations related to each of the aircraft components. The equations and notation used here are from (RA92).

Wing weight

$$W_{wing} = 0.036 S_w^{\,0.758} W_{fw}^{\,0.0035} \left(\frac{AR}{\cos^2 \Lambda} \right)^{0.6} q_D^{\,0.006} \lambda^{0.04} \left(\frac{100 t/c}{\cos \Lambda} \right)^{-0.3} \left(N_z W_{dg} \right)^{0.49} \tag{3-2}$$

Horizontal tail weight

$$W_{\substack{horizontal\\tail}} = 0.016\left(N_z W_{dg}\right)^{0.414} q_D{}^{0.168} S_{ht}{}^{0.896} \left(\frac{100t/c}{\cos\Lambda}\right)^{-0.12} \left(\frac{AR}{\cos^2\Lambda_{ht}}\right)^{0.043} \lambda_{ht}{}^{-0.02} \tag{3-3}$$

Vertical tail weight

$$W_{\substack{vertical\\tail}} = 0.073\left(1+0.2\frac{H_t}{H_v}\right)\left(N_z W_{dg}\right)^{0.376} q_D{}^{0.122} S_{vt}{}^{0.873} \left(\frac{100t/c}{\cos\Lambda_{vt}}\right)^{-0.49} \left(\frac{AR}{\cos^2\Lambda_{vt}}\right)^{0.357} \lambda_{vt}{}^{0.039} \tag{3-4}$$

Fuselage weight

$$W_{fuselage} = 0.052 S_f{}^{1.086} \left(N_z W_{dg}\right)^{0.177} L_t{}^{-0.051} \left(\frac{L_B}{W_{fus}}\right)^{-0.072} q_D{}^{0.241} + W_{press} \tag{3-5a}$$

where the weight penalty due to pressurization is:

$$W_{press} = 11.9 + \left(V_{pr} P_{delta}\right)^{0.271} \tag{3-5b}$$

Main landing gear weight

$$W_{\substack{main\text{-}landing\\gear}} = 0.095\left(N_l W_l\right)^{0.768} \left(\frac{L_m}{12}\right)^{0.409} \tag{3-6}$$

Nose landing gear weight

$$W_{\substack{nose\cdot landing \\ gear}} = 0.125(N_l W_l)^{0.566}\left(\frac{L_n}{12}\right)^{0.845} \tag{3-7}$$

Total installed engine weight

$$W_{\substack{installed\cdot engine \\ (total)}} = 2.575 W_{en}^{\ 0.922} N_{en} \tag{3-8}$$

Fuel system weight

This equation does not include the actual fuel weight.

$$W_{fuel\cdot system} = 2.49 V_t^{\ 0.726}\left(\frac{1}{1+V_i/V_t}\right)^{0.363} N_t^{\ 0.242} N_{en}^{\ 0.157} \tag{3-9}$$

Flight controls weight

$$W_{\substack{flight \\ controls}} = 0.053 L_B^{\ 1.536} b^{0.371}\left(N_z W_{dg} \times 10^{-4}\right)^{0.80} \tag{3-10}$$

Hydraulics system weight

$$W_{hydraulics} = 0.001 W_{dg} \tag{3-11}$$

Avionics system weight

$$W_{avionics} = 2.114 W_{uav}^{0.933} \tag{3-12}$$

Electrical system weight

$$W_{electrical} = 12.57\left(W_{fuel\cdot system} + W_{avionics}\right)^{0.51} \tag{3-13}$$

Air conditioning and anti-ice system weight

$$W_{\substack{air\cdot conditioning \\ and\cdot anti\cdot ice}} = 0.265 W_{dg}^{0.52} N_p^{\ 0.68} W_{avionics}^{0.17} M^{0.08} \tag{3-14}$$

Furnishings weight

$$W_{furnishings} = 0.0582 W_{dg} - 65 \tag{3-15}$$

3.3 Center of Gravity and Moments of Inertia

The center of gravity location is determined using the weight and x-coordinate position of each aircraft component. The equation used is:

$$X_{C.G._{aircraft}} = \frac{\sum X_{C.G_{component}} W_{component}}{\sum W_{component}} \tag{3-16}$$

The aircraft moments of inertia must be known in order to perform the dynamic stability analysis. Using statistical equations these moments of inertia are determined as a function of the aircraft nondimensional radii of gyration (RO85). They are given by (RA92) as:

$$I_{xx} = \frac{b^2 W_{total} R_x^{\ 2}}{4g} \tag{3-17}$$

$$I_{yy} = \frac{L_B^{\ 2} W_{total} R_y^2}{4g} \tag{3-18}$$

$$I_{zz} = \left(\frac{b + L_B}{2}\right)^2 \frac{W_{total} R_z^2}{4g} \tag{3-19}$$

where the radii of gyration are obtained from statistical data for similar airplanes. At the same time, using the location x-coordinates and weights of all components, the moments of inertia can be calculated directly by summation of component contributions. The two estimates can be compared. Any one of them or their average values can be used to evaluate dynamic stability and control characteristics.

4. AERODYNAMICS FOR MISSION PERFORMANCE

The aerodynamic characteristics of each flight mission phase must be determined appropriately. The following chapter focuses on the assumptions and calculations made in developing the lift and drag coefficients to be used in the performance equations for mission analysis.

4.1 Takeoff

Under the assumption (RA92) that $U_{takeoff} = 1.2U_{stall}$ the lift coefficient at take-off is:

$$C_{L_{takeoff}} = \frac{C_{L_{\max}}}{(1.2)^2} \tag{4-1}$$

The drag coefficient at takeoff is modified from the standard drag polar calculation by adding ground effect at takeoff and effects of the extension of the landing gear and flaps. This becomes (TO82):

$$C_{D_{takeoff}} = C_{D_0} + \frac{C_{L_{takeoff}}^2}{\pi ARS} g_{effect} + \frac{\Delta C_{D_{gear}}}{\Delta C_{D_0}} C_{D_0} + \frac{\Delta C_{D_{Flap}}}{\Delta C_{D_0}} C_{D_0} \tag{4-2}$$

4.2 Climb

For the climb of a propeller driven airplane, (RA92) defines the optimum lift coefficient to be

$$C_{L_{climb}} = \sqrt{3\pi eARC_{D_0}} \tag{4-3}$$

The drag coefficient is found using the standard method for drag polar calculation:

$$C_D = C_{D_0} + \frac{C_L^2}{\pi e AR} \tag{4-4}$$

4.3 Cruise

At cruise, (TO82) defines the optimum lift coefficient to be

$$C_{L_{cruise}} = C_{L_{fraction}} \sqrt{\pi e AR C_D} \tag{4-5}$$

The $C_{L_{fraction}}$ term is added to allow for a more conservative estimate of the actual lift coefficient encountered at altitude. The drag coefficient is calculated using Equation 4-4.

4.4 Loiter

The lift and drag coefficients for loiter are calculated in the same manner as those for the climb leg.

4.5 Landing

Under the assumption presented by (RA92) that $U_{landing} = 1.21 U_{stall}$ the lift coefficient is:

$$C_{L_{landing}} = \frac{C_{L_{max}}}{(1.21)^2} \tag{4-6}$$

The performance calculations also require an estimation of the lift coefficient on the runway, which is assumed to be roughly 80% of the landing lift coefficient. The drag coefficient is determined by Equation 4-2.

5. MISSION PERFORMANCE

This chapter outlines the analysis conducted in determining the mission performance of the aircraft. Of primary concern are issues of range, loiter, landing and takeoff distances, rate of climb, power, and mission fuel burn.

5.1 Power available versus power required

With the exception of landing, each mission leg calculates the power available and the power required. This information is used in the set of constraints to guarantee that the airplane has enough power at all mission segments. This is primarily a function of the max power of the engine(s) and the altitude:

$$P_{available} = P_{\max}\left(\frac{\rho}{\rho_{SL}}\right)^{A_p} \eta_p \cdot throttle \tag{5-1}$$

where Ap is used to model the variation of engine power with flight altitude, and "throttle" is used to limit the throttle allowed for each mission segment. The power required is simply drag times velocity. These parameters are determined by using the appropriate lift or drag coefficient for the mission leg in question (assuming one-g flight):

$$U = \sqrt{\frac{2W}{\rho S C_L}} \tag{5-2}$$

$$Drag = 0.5\rho U^2 S C_D \tag{5-3}$$

5.2 Rate of climb

The rate of climb is calculated for the takeoff, climb, and cruise mission legs. It is strongly dependent on the excess power. The equation implemented is for the optimum rate of climb (RA92) and is given as:

$$ROC = \frac{P_{available}}{W} - \frac{\rho U^3 C_{D_0}}{2(W/S)} - \frac{2(W/S)}{\pi e AR \rho} \tag{5-4}$$

The climb leg actually implements an alternate form of the equation determining the rate of climb.

$$\frac{dh}{dt} = \frac{\left(P_{available} - P_{required}\right)}{W} \tag{5-5}$$

5.3 Balanced field length

The balanced field length is the total takeoff distance including obstacle clearance that is needed to allow the aircraft to brake to a halt or takeoff in the event of an engine failure (RA92). The equation used for this calculation is (TO82):

$$BFL = 0.863\left(\frac{W/S}{\rho g C_{L_{takeoff}}} + h_{obstacle}\right)\left(\frac{1}{T_{av}/W - U_r} + 2.7\right) + \left(\frac{655}{\sqrt{(\rho/\rho_{SL})}}\right) \tag{5-6a}$$

where the average thrust is calculated

$$T_{av} = 5.75 P_{available}\left[\frac{(\rho/\rho_{SL}) N_{en} D_{prop}^2}{P_{available}}\right]^{1/3} \tag{5-6b}$$

and

$$U_r = 0.01C_{L_{max}} + 0.02 \tag{5-6c}$$

5.4 Landing distance

The landing distance is calculated based on a method presented in (TO82), which approximates the touchdown speed based on an estimated landing flare. The total equation is presented as:

$$S_{land} = \left[\frac{1}{\bar{\gamma}} + 1.69\left[\frac{1}{\bar{a}/g}\left(1 - \frac{\bar{\gamma}^2}{\Delta n}\right) + \frac{\bar{\gamma}}{\Delta n}\right]\left(\frac{W/S}{h_{land}\rho g C_{L_{max}}}\right)\right] h_{land} \tag{5-7a}$$

where the average deceleration is calculated from the drag and the friction force between the aircraft and the runway

$$\bar{a} = \frac{\left[Drag + (W - Lift)\mu_{runway}\right]}{W} g \tag{5-7b}$$

5.5 Mission fuel burn

The mission fuel burn acts as the main iterative variable when it comes to determining the actual weight of an aircraft design. The methods used for each leg follow in terms of the final weight versus the initial weight.

Takeoff

$$W_{final} = 0.97 W_{initial} \qquad \text{(RA92)} \tag{5-8}$$

Climb

Two options exist in calculating fuel burn for the climb segment. The first is statistical:

$$W_{final} = 0.985 W_{initial} \qquad \text{(RA92)} \qquad (5\text{-}9)$$

The second is performance based:

$$W_{final} = W_{initial} - \left(P_{available} BSFC\right) dt \qquad (5\text{-}10a)$$

where

$$dt = \frac{\Delta h}{dh/dt} \qquad (5\text{-}10b)$$

Cruise

$$W_{final} = W_{initial} \exp\left[\frac{-Range \cdot BSFC}{550 \eta_p (L/D)_{\max}}\right] \qquad \text{(RA92)} \qquad (5\text{-}11)$$

Loiter

$$W_{final} = W_{initial} \exp\left[\frac{-Endurance \cdot U \cdot BSFC}{550 \eta_p (L/D)_{\max}}\right] \qquad \text{(RA92)} \qquad (5\text{-}12)$$

Landing

$$W_{final} = 0.995 W_{initial} \qquad \text{(RA92)} \qquad (5\text{-}13)$$

When the mission analysis concludes (starting with a guess of the gross take-off weight and ending with the final weight for the mission, RA92, RO85), the fuel weight used is found as:

$$W_{fuelburn} = W_{guess} - W_{final} \qquad (5\text{-}14)$$

5.6 Mission weight analysis

Aside from the component-by-component weight analysis of Chapter 3, another weight analysis occurs within the mission analysis based on statistical equations for the ratio of the overall empty weight to the take-off gross weight of the airplane. This calculates the empty weight from the mission analysis as:

$$W_{empty} = W_{guess} - W_{final} - W_{crew} - W_{passengers} - W_{payload} \tag{5-15}$$

This empty weight is compared to the empty weight calculated using the average of two alternate methods:

$$W_{empty1} = 10\text{^}\left[\frac{\ln W_{guess} - RoskamA}{RoskamB}\right] \qquad \text{(RO85)} \tag{5-16}$$

and

$$W_{empty2} = W_{guess} RaymerA \cdot W_{guess}^{RaymerC} \qquad \text{(RA92)} \tag{5-17}$$

In conventional sizing of an airplane during conceptual design the weight is arrived at by an iterative process in which a guess of the take-off gross weight is improved iteratively (RA92) or by solving directly for the take-off gross weight (RO85). A weight guess is used to start a mission analysis in which weight is reduced (due to fuel burn) as the airplane progresses through the mission. When the end weight is obtained, a new estimate of the take-off weight can be calculated. In the present capability, as will be described later, the matching of the weight estimate at the end of a mission run and the input guess weight which starts the mission analysis is done by imposing an equality constraint and

counting on the optimization algorithm used to meet this constraint.

The gross weight at the end of a mission is calculated from:

$$W_{gross} = \frac{W_{crew} + W_{passengers} + W_{payload}}{1 - \left(W_{fuel}/W_{guess}\right) - \left(W_{empty_{average}}/W_{guess}\right)} \qquad \text{(RA92)} \qquad (5\text{-}18)$$

5.7 Mission segments

The mission is defined to match desired mission requirements such as takeoff distance, rate of climb, range and endurance. These parameters are presented below.

TABLE 5.7.1: Mission segments and their defining variables

NAME	DESCRIPTION
altitude	Altitude or change in altitude (ft)
Dist	Distance or endurance (miles or minutes)
BSFC (lb/hr/hp)	Brake specific fuel consumption (lb/hr/hp)
Np	Propeller efficiency
Speed	Actual airspeed or required airspeed (ft/s)
throttle	Throttle setting
Runway(TO/LD)	Takeoff or landing runway length (ft)
ROC	Required rate of climb (ft/min)
cLfraction	Lift coefficient correction factor
cLratio	Lift coefficient correction factor
gammabar	Used for Torenbeek approximation
deltaN	Used for Torenbeek approximation
groundfriction	Ground friction coefficient
Hobstacle	Obstacle height at takeoff or landing (ft)

6. GEOMETRY FOR STABILITY AND CONTROL AERODYNAMICS

Definition of geometric pre-assigned parameters and variations of the geometric design variables describing a configuration lead to variations in the aerodynamic characteristics affecting stability and control. The present chapter lists a number of geometric variables and their uses in static and dynamic stability and control analysis.

6.1 Mean Aerodynamic Chord

The mean aerodynamic chord is used for all calculations pertaining to pitching moment coefficients in longitudinal stability derivatives. For a trapezoidal wing planform (SM84, TO82) the mean aerodynamic chord is:

$$\bar{c} = \frac{2}{3} c_r \frac{\left(1+\lambda+\lambda^2\right)}{\left(1+\lambda\right)} \tag{6-1}$$

6.2 Wetted Area

The wetted area of the aircraft is required for two reasons: 1.) It allows the parasite drag coefficient to be estimated (based on equivalent flat plate friction coefficients for comparable configurations) for each new design encountered during optimization. Thus, relative size changes of wing and overall size of the vehicle can affect the zero-lift drag coefficient C_{Do}; 2.) Weights in the weight analysis presented in Chapter 1 are dependent on the fuselage's wetted area.

The approximation presented in (RA92) divides the aircraft into four sections: fuselage, wing, horizontal tail, and vertical tail. The fuselage's wetted area is determined by:

$$S_f = 3.4\left(\frac{A_{top} + A_{side}}{2}\right) \tag{6-2a}$$

where the fuselage width and height are taken as an average of known values such that:

$$A_{top} = \frac{w_{nose} + w_{fcy} + w_{bcy} + w_{fus}}{4} L_B \tag{6-2b}$$

$$A_{side} = \frac{h_{nose} + h_{fcy} + h_{bcy}}{3} L_B \tag{6-2c}$$

The wetted area of the wing is calculated as:

$$S_{wet_{wing}} = S_{ex\cdot posed}\left(1.971 + 0.52(t/c)\right) \tag{6-3a}$$

where:

$$S_{ex\cdot posed} = S - c_{root} w_{fus} \tag{6-3b}$$

The wetted area of the horizontal tail and vertical tail is taken to be twice the projected planform area of the respective surfaces. The thickness of these surfaces is assumed very small.

6.3 Parasite Drag Coefficient

The parasite drag coefficient is determined using the flat plate equivalent skin friction coefficient and wetted area as outlined in (CO85,PH49). The relationship is given by

$$C_{D_o} = \frac{S_{wet}}{S} C_F \tag{6-4}$$

where S is the projected area of the wing used to non-dimensionalize all aerodynamic force and moment coefficients.

6.4 Induced angle of attack

The induced angle of attack due to lift on the wing contributes to the effective wing angle of attack. This same effect occurs at the horizontal tail. Taken from (DSC67), these effects are determined by

$$\alpha_i = \frac{C_L}{\pi e_1 AR} \tag{6-5}$$

This equation introduces the induced-angle span efficiency factor, which is calculated as

$$e_1 = \frac{1}{1+\tau} \tag{6-6a}$$

where the correction factor, τ, depends on the taper ratio and the aspect ratio.

For the case of taper ratio $\lambda = 1$,

$$\tau = 0.0000297115AR^4 - 0.000811747AR^3 + 0.0071717AR^2 - 0.00298853AR + 0.07739 \tag{6-6b}$$

For those cases where $\lambda \neq 1$, the following equation holds only for aspect ratio values around 6.28.

$$\tau = 6526.0\lambda^{13} - 33936.5\lambda^{12} + 65708.3\lambda^{11} - 40275.9\lambda^{10} - 55022.8\lambda^9 + 134075.0\lambda^8 - 128452.0\lambda^7 + 71152.4\lambda^6 - 24169.6\lambda^5 + 4910.63\lambda^4 - 541.19\lambda^3 + 27.1966\lambda^2 - 1.31155\lambda + 0.170212 \tag{6-6c}$$

6.5 Oswald Coefficient

Within the mission performance module the Oswald coefficient is set by the user. In this case, a correction factor has been added to account for non-elliptic lift distributions on the wing planform (PH49). This corrected coefficient is calculated as:

$$e = \frac{1}{1+\delta} \tag{6-7a}$$

where the correction factor, δ, depends on the taper ratio and the aspect ratio.

For the case of taper ratio $\lambda = 1$,

$$\delta = -0.0000143113AR^3 + 0.0000158924AR^2 + 0.0113969AR - 0.011339 \tag{6-7b}$$

For those cases where $\lambda \neq 1$, the following equation is valid for aspect ratio values around 6.0.

$$\delta = 2.48637\lambda^6 - 9.29906\lambda^5 + 14.0692\lambda^4 - 11.1199\lambda^3 + 5.01493\lambda^2 - 1.24262\lambda + 0.141122 \tag{6-7c}$$

6.6 Downwash on the Tail

The angle of attack of the horizontal tail is determined by the wing's angle of attack, the wing's incidence angle, the tail's incidence angle, and the downwash on the tail due to the wing. The change in tail downwash with respect to the wing angle of attack is given as (DSC67):

$$\frac{d\varepsilon}{d\alpha} = 20C_{L_\alpha}\frac{(1/\lambda)^{0.3}}{AR^{0.725}}\left(\frac{3\bar{c}}{l_t'}\right) \tag{6-8}$$

6.7 Effective aspect ratio of the vertical tail

The effective aspect ratio of the vertical tail is needed to calculate the respective 3-D lift curve slope. This is often greater than the geometric aspect ratio for vertical tails mounted above the horizontal tail and is found to be (PH49):

$$AR_{eff_{vert}} = 1.55\frac{b_v^2}{S_v} \tag{6-9}$$

$AR_{eff_{vert}}$ is limited to values equal to or less than 6.5 for the purpose of this analysis (SM84).

6.8 Body Reference Area

The reference body area is used to determine fuselage effects on lateral-directional stability. This is calculated as a function of the fuselage volume

$$A_{bodyref} = V_{fuse}^{2/3} \tag{6-10}$$

The fuselage volume is estimated using the volume of four prismoids derived from the fuselage geometry (SM84). These relationships are:

$$V_{fuse} = \frac{1}{6}(V_1 + V_2 + V_3 + V_4) \tag{6-11a}$$

$$V_1 = l_{fcy}\left[2\left(h_{nose}w_{nose} + h_{fcy}w_{fcy}\right) + h_{fcy}w_{nose} + h_{nose}w_{fcy}\right] \tag{6-11b}$$

$$V_2 = \left(l_{mh} - l_{fcy}\right)\left[2\left(h_{fcy}w_{fcy} + hw_{fus}\right) + hw_{fcy} + h_{fcy}w_{fus}\right] \quad (6\text{-}11c)$$

$$V_3 = \left(l_{bcy} - l_{mh}\right)\left[2\left(h_{bcy}w_{bcy} + hw_{fus}\right) + hw_{bcy} + w_{fus}h_{bcy}\right] \quad (6\text{-}11d)$$

$$V_4 = \left(L_B - l_{bcy}\right)\left[2\left(h_{bcy}w_{bcy} + 4r_1^2\right) + 2h_{bcy}r_1 + 2w_{bcy}r_1\right] \quad (6\text{-}11e)$$

6.9 Side Body Area

Introducing the side body area is one method of adding a wing-fuselage correction in the determination of static directional stability. This area is estimated using the following relationship from (SM84):

$$S_{B_s} = \left(h_{nose} + h_{fcy}\right)\frac{l_{fcy}}{2} + \left(h + h_{fcy}\right)\frac{\left(l_{mh} - l_{fcy}\right)}{2} + \left(h + h_{bcy}\right)\frac{\left(l_{bcy} - l_{mh}\right)}{2} + \left(h_{bcy} + 2r_1\right)\frac{\left(L_B - l_{bcy}\right)}{2} \quad (6\text{-}12)$$

7. AERODYNAMIC STABILITY AND CONTROL DERIVATIVES

This chapter presents the analysis methods used for determining all stability and control derivatives, non-dimensional and dimensional. Longitudinal derivatives are presented first followed by lateral-directional derivatives.

7.1 Non-dimensional longitudinal stability derivatives

The following derivatives apply to translational motion along the x-body-axis and rotational motion in pitch about the y-body-axis. Unless otherwise noted, these calculations based on (SM84) which, in turn, is a compilation of approximate equations based on (PH49) and the NACA reports listed in the bibliography section. As indicated by the title of this section, all rate derivatives (rate of change of angle of attack and pitch rate) are dimensionless.

7.1.1 Aircraft lift coefficient

The total aircraft lift coefficient is calculated using the standard relationship

$$C_L = \frac{W}{q_D S} \tag{7-1a}$$

Dividing the lift coefficient into wing and horizontal tail contributions yields,

$$C_L = C_{L_{wing}} + C_{L_{tail}} \frac{S_t}{S} \eta_t \tag{7-1b}$$

where the wing contribution is determined as:

$$C_{L_{wing}} = \frac{C_L}{1 + \frac{x_a S}{l_t S_t}\eta_t} \tag{7-1c}$$

And the contribution due to the horizontal tail is calculated by

$$C_{L_{tail}} = \frac{C_{L_{wing}} x_a S}{l_t S_t \eta_t} \tag{7-1d}$$

7.1.2 Aircraft drag coefficient

The total aircraft drag coefficient is calculated using the standard drag polar equation

$$C_D = C_{D_0} + \frac{C_L^2}{\pi e AR} \tag{7-2}$$

7.1.3 Pitching moment coefficient

In equilibrium flight, the pitching moment coefficient is zero unless the thrust line is some distance z_j from the center of gravity. In this case the aerodynamic moment has to balance the moment due to thrust about the center of gravity. Assuming the thrust equals the drag in unaccelerated flight, the moment coefficient becomes

$$C_m = \frac{C_D z_j}{\bar{c}} \tag{7-3}$$

7.1.4 Thrust coefficient

Though not explicitly utilized in the present power-off stability analysis, the thrust coefficient is calculated as

$$C_T = \frac{Thrust}{qS} = C_D \tag{7-4}$$

7.1.5 Change in lift coefficient with angle of attack

Though it is possible to separate the wing and horizontal tail contributions to this stability derivative, only the wing contribution is used in this analysis, assuming a conventional design with the wing as the major lift generating element. The lift-curve-slope, as this derivative is often called, is given as

$$C_{L_\alpha} = \frac{C_{l_\alpha}}{1 + 57.3\dfrac{C_{l_\alpha}}{\pi e_1 AR}} \tag{7-5}$$

(Note the dependency on units selected here. The lift curve slope is in 1/degree)

The tail contribution is calculated for its role in the determination of C_{m_α}. The corresponding relationship follows logically as:

$$C_{L_{\alpha_{tail}}} = \frac{C_{l_{\alpha_{tail}}}}{1 + 57.3\dfrac{C_{l_{\alpha_{tail}}}}{\pi e_{1_t} AR_t}} \tag{7-6}$$

7.1.6 Change in drag coefficient with angle of attack

The wing, fuselage, and tail section all contribute to C_{D_α}. However, assuming small perturbations, the effects due to the fuselage and the tail are ignored. This derivative is calculated by

$$C_{D_\alpha} = C_{d_\alpha} + \frac{2C_L}{\pi e AR} C_{L_\alpha} \tag{7-7}$$

7.1.7 Change in pitching moment coefficient with angle of attack

The contributions of the wing, horizontal tail, and fuselage are included in the calculation of C_{m_α}. This equation, written as the sum of its components, is:

$$C_{m_\alpha} = C_{m_{\alpha\,wing}} + C_{m_{\alpha_{tail}}} + C_{m_{\alpha\,fuselage}} \tag{7-8a}$$

where,

$$C_{m_{\alpha_{wing}}} = C_{L_\alpha}\left[\left(1 + \frac{2C_L}{57.3\pi e AR}(\alpha - i_{wing}) + \frac{C_D}{C_{L_\alpha}}\right)\frac{x_a}{\bar{c}} + \left(\frac{2C_L}{\pi e AR} - \frac{(\alpha - i_{wing})}{57.3} - \frac{C_L}{C_{L_\alpha}}\right)\frac{z_a}{\bar{c}}\right] \tag{7-8b}$$

$$C_{m_{\alpha_{tail}}} = -C_{L_{\alpha_{tail}}}\left(1 - \frac{d\varepsilon}{d\alpha}\right)\frac{S_t}{S}\frac{l_t}{\bar{c}}\eta_t \qquad \text{(DSC67)} \tag{7-8c}$$

and

$$C_{m_{\alpha\,fuselage}} = \frac{k_f w_f^2 L_B}{S\bar{c}} \qquad \text{(GW41)} \tag{7-8d}$$

The k_f term is an empirical factor used in the estimation of fuselage and nacelle effect on C_{m_α}. This factor is found to be (PH49, SM84)

$$k_f = -19.6558\left(\frac{x_{fus}}{L_B}\right)^4 + 50.3465\left(\frac{x_{fus}}{L_B}\right)^3 - 26.5479\left(\frac{x_{fus}}{L_B}\right)^2 + 7.47299\left(\frac{x_{fus}}{L_B}\right) - 0.464064 \tag{7-8e}$$

7.1.8 Change in lift coefficient with rate of change of angle of attack

This derivative results primarily from the time lag effect at the horizontal tail due to downwash from the wing and is computed as:

$$C_{L_{\dot{\alpha}}} = 2C_{L_{\alpha_{tail}}} \frac{d\varepsilon}{d\alpha} \frac{l_{t_1}}{\bar{c}} \frac{S_t}{S} \eta_t \tag{7-9}$$

7.1.9 Change in drag coefficient with rate of change of angle of attack

The change in drag coefficient with rate of change of angle of attack is negligible for the class of aircraft with which this research is concerned. This derivative is set as zero for all analysis.

7.1.10 Change in pitching moment coefficient with rate of change of angle of attack

This derivative represents the change in pitching moment coefficient with rate of change of angle of attack. (GS44) presents an expression for this derivative in the form

$$C_{m_{\dot{\alpha}}} = -2C_{L_{\alpha_{tail}}} \frac{d\varepsilon}{d\alpha} \frac{l_t}{\bar{c}} \frac{l_{t_1}}{\bar{c}} \frac{S_t}{S} \eta_t \tag{7-10}$$

7.1.11 Change in lift coefficient with pitch rate

The main contribution to this derivative comes from the horizontal tail. The following equation from (DU63) calculates the derivative as:

$$C_{L_q} = 2\frac{l_t}{\bar{c}} C_{L_{\alpha_{tail}}} \frac{S_t}{S} \eta_t \tag{7-11}$$

7.1.12 Change in drag coefficient with pitch rate

While both the wing and fuselage contribute to this derivative, their magnitudes are very small and are neglected. This derivative is set as zero the purpose of this analysis.

7.1.13 Change in pitching moment coefficient with pitch rate

This derivative is sometimes referred to as the "pitch damping" derivative. The wing contribution is small compared to the contribution due to the horizontal tail. The fuselage contribution is even more negligible and is always omitted for light aircraft. The equation for this derivative is given as

$$C_{m_q} = -2\frac{x'}{\bar{c}^2}|x'|C_{L_\alpha} - 2\frac{l_t^2}{\bar{c}^2} C_{L_{\alpha_{tail}}} \frac{S_t}{S} \eta_t \tag{7-12}$$

7.1.14 Change in lift coefficient due to elevator deflection

This derivative quantifies the change in the lift coefficient with deflection of the elevator. The following relationship from (HE68) is used to calculate this derivative.

$$C_{L_{\delta_E}} = C_{l_{\delta_E}} \frac{C_{L_{\alpha_{tail}}}}{C_{l_{\alpha_{tail}}}} \frac{(\alpha_\delta)_{C_L}}{(\alpha_\delta)_{C_l}} \frac{S_t}{S} \eta_t \tag{7-13}$$

The new terms presented in equation 7-13a are calculated using data presented in (DSC67). The first term, $C_{l_{\delta_E}}$, is determined as a function of the ratio of the elevator chord to the chord of the horizontal tail and on the aspect ratio of the tail. The corresponding equations are summarized (SM84):

For the case where, $\frac{c_{elev}}{\bar{c}_{tail}} \leq 0.4$, no restriction on AR_t exists and,

$$C_{l_{\delta_E}} = -0.0580767\left(\frac{c_{elev}}{\bar{c}_{tail}}\right)^4 + 0.146101\left(\frac{c_{elev}}{\bar{c}_{tail}}\right)^3 - 0.169823\left(\frac{c_{elev}}{\bar{c}_{tail}}\right)^2 + 0.194084\left(\frac{c_{elev}}{\bar{c}_{tail}}\right) - 0.00202833 \tag{7-14a}$$

This equation also holds for the case where $\frac{c_{elev}}{\bar{c}_{tail}} > 0.4$ and $AR_t > 4.5$.

When $\frac{c_{elev}}{\bar{c}_{tail}} > 0.4$ and $AR_t \leq 4.5$,

$$C_{l_{\delta_E}} = 0.0563 + 0.0321\left(\frac{c_{elev}}{\bar{c}_{tail}} - 0.4\right) \tag{7-14b}$$

The next term is the two-dimensional flap effectiveness parameter, $(\alpha_\delta)_{C_l}$. This is calculated as:

$$(\alpha_\delta)_{C_l} = 4.65842\left(\frac{c_{elev}}{\overline{c}_{tail}}\right)^4 - 10.9873\left(\frac{c_{elev}}{\overline{c}_{tail}}\right)^3 + 9.47521\left(\frac{c_{elev}}{\overline{c}_{tail}}\right)^2 - 4.09969\left(\frac{c_{elev}}{\overline{c}_{tail}}\right) - 0.0432959 \quad (7\text{-}15)$$

The value for $(\alpha_\delta)_{C_l}$ directly effects the next series of calculations to find a value for the ratio of the three-dimensional flap effectiveness parameter to the two-dimensional flap effectiveness parameter. These calculations are summarized as:

For $(\alpha_\delta)_{C_l} \geq -0.1$:

$$\frac{(\alpha_\delta)_{C_L}}{(\alpha_\delta)_{C_l}} = -0.0000580764AR_t^5 + 0.00203746AR_t^4 - 0.0284903AR_t^3 + 0.203783AR_t^2 - 0.802715AR_t + 2.77199 \quad (7\text{-}16a)$$

For $-0.2 \leq (\alpha_\delta)_{C_l} < -0.1$:

$$\frac{(\alpha_\delta)_{C_L}}{(\alpha_\delta)_{C_l}} = (\alpha_\delta)_1 + \left((\alpha_\delta)_2 - (\alpha_\delta)_1\right)\left(\frac{(\alpha_\delta)_{C_l} + 0.1}{-0.1}\right) \quad (7\text{-}16b)$$

$$(\alpha_\delta)_1 = -0.0000580764AR_t^5 + 0.00203746AR_t^4 - 0.0284903AR_t^3 + 0.203783AR_t^2 - 0.802715AR_t + 2.77199 \quad (7\text{-}16c)$$

$$(\alpha_\delta)_2 = -0.000858209AR_t^3 + 0.0224724AR_t^2 - 0.206709AR_t + 1.81719 \quad (7\text{-}16d)$$

For $-0.3 \le (\alpha_\delta)_{C_l} < -0.2$:

$$\frac{(\alpha_\delta)_{C_L}}{(\alpha_\delta)_{C_l}} = (\alpha_\delta)_1 + \left((\alpha_\delta)_2 - (\alpha_\delta)_1\right)\left(\frac{(\alpha_\delta)_{C_l} + 0.2}{-0.1}\right) \tag{7-16e}$$

$$(\alpha_\delta)_1 = -0.000858209AR_t^3 + 0.0224724AR_t^2 - 0.206709AR_t + 1.81719 \tag{7-16f}$$

$$(\alpha_\delta)_2 = 0.000132994AR_t^4 - 0.00394386AR_t^3 + 0.0450323AR_t^2 - 0.249244AR_t + 1.69816 \tag{7-16g}$$

For $-0.4 \le (\alpha_\delta)_{C_l} < -0.3$:

$$\frac{(\alpha_\delta)_{C_L}}{(\alpha_\delta)_{C_l}} = (\alpha_\delta)_1 + \left((\alpha_\delta)_2 - (\alpha_\delta)_1\right)\left(\frac{(\alpha_\delta)_{C_l} + 0.3}{-0.1}\right) \tag{7-16h}$$

$$(\alpha_\delta)_1 = 0.000132994AR_t^4 - 0.00394386AR_t^3 + 0.0450323AR_t^2 - 0.249244AR_t + 1.69816 \tag{7-16i}$$

$$(\alpha_\delta)_2 = -0.00000724351AR_t^6 + 0.000235239AR_t^5 - 0.00285365AR_t^4 + 0.0150882AR_t^3 - 0.0214656AR_t^2 - 0.108685AR_t + 1.47429 \tag{7-16j}$$

For $-0.5 \le (\alpha_\delta)_{C_l} < -0.4$:

$$\frac{(\alpha_\delta)_{C_L}}{(\alpha_\delta)_{C_l}} = (\alpha_\delta)_1 + \left((\alpha_\delta)_2 - (\alpha_\delta)_1\right)\left(\frac{(\alpha_\delta)_{C_l} + 0.4}{-0.1}\right) \tag{7-16k}$$

$$(\alpha_\delta)_1 = -0.00000724351AR_t^6 + 0.000235239AR_t^5 - 0.00285365AR_t^4 + 0.0150882AR_t^3 - 0.0214656AR_t^2 - 0.108685AR_t + 1.47429 \tag{7-16l}$$

$$(\alpha_\delta)_2 = -0.000011681AR_t^5 + 0.000430597AR_t^4 - 0.00621814AR_t^3 + 0.0456723AR_t^2 - 0.184917AR_t + 1.41226 \tag{7-16m}$$

For $-0.6 \le (\alpha_\delta)_{C_l} < -0.5$:

$$\frac{(\alpha_\delta)_{C_L}}{(\alpha_\delta)_{C_l}} = (\alpha_\delta)_1 + \left((\alpha_\delta)_2 - (\alpha_\delta)_1\right)\left(\frac{(\alpha_\delta)_{C_l} + 0.5}{-0.1}\right) \tag{7-16n}$$

$$(\alpha_\delta)_1 = -0.000011681 AR_t^5 + 0.000430597 AR_t^4 - 0.00621814 AR_t^3 + 0.0456723 AR_t^2 - 0.184917 AR_t + 1.41226 \tag{7-16o}$$

$$(\alpha_\delta)_2 = -0.0000073708 AR_t^5 + 0.000307952 AR_t^4 - 0.00489224 AR_t^3 + 0.0380384 AR_t^2 - 0.154526 AR_t + 1.32116 \tag{7-16p}$$

For $-0.7 \le (\alpha_\delta)_{C_l} < -0.6$:

$$\frac{(\alpha_\delta)_{C_L}}{(\alpha_\delta)_{C_l}} = (\alpha_\delta)_1 + \left((\alpha_\delta)_2 - (\alpha_\delta)_1\right)\left(\frac{(\alpha_\delta)_{C_l} + 0.6}{-0.1}\right) \tag{7-16q}$$

$$(\alpha_\delta)_1 = -0.0000073708 AR_t^5 + 0.000307952 AR_t^4 - 0.00489224 AR_t^3 + 0.0380384 AR_t^2 - 0.154526 AR_t + 1.32116 \tag{7-16r}$$

$$(\alpha_\delta)_2 = 0.00000243494 AR_t^4 - 0.000255415 AR_t^3 + 0.0057768 AR_t^2 - 0.048723 AR_t + 1.16569 \tag{7-16s}$$

For $-0.8 \le (\alpha_\delta)_{C_l} < -0.7$:

$$\frac{(\alpha_\delta)_{C_L}}{(\alpha_\delta)_{C_l}} = (\alpha_\delta)_1 + \left((\alpha_\delta)_2 - (\alpha_\delta)_1\right)\left(\frac{(\alpha_\delta)_{C_l} + 0.7}{-0.1}\right) \tag{7-16t}$$

$$(\alpha_\delta)_1 = 0.00000243494 AR_t^4 - 0.000255415 AR_t^3 + 0.0057768 AR_t^2 - 0.048723 AR_t + 1.16569 \tag{7-16u}$$

$$(\alpha_\delta)_2 = 0.0000255244 AR_t^4 - 0.00080707 AR_t^3 + 0.00935254 AR_t^2 - 0.0487466 AR_t + 1.12006 \tag{7-16v}$$

And for $-1.0 \le (\alpha_\delta)_{C_l} < -0.8$:

$$\frac{(\alpha_\delta)_{C_{L^*}}}{(\alpha_\delta)_{C_l}} = (\alpha_\delta)_1 + \left((\alpha_\delta)_2 - (\alpha_\delta)_1\right)\left(\frac{(\alpha_\delta)_{C_l} + 0.8}{-0.2}\right) \tag{7-16w}$$

$$(\alpha_\delta)_1 = 0.0000255244 AR_t^4 - 0.00080707 AR_t^3 + 0.00935254 AR_t^2 - 0.0487466 AR_t + 1.12006 \tag{7-16x}$$

$$(\alpha_\delta)_2 = 1.0 \tag{7-16y}$$

7.1.15 Change in drag coefficient due to elevator deflection

Assuming a reasonably sized elevator, this derivative has very little effect on the total airplane drag and is therefore neglected for the purposes of this analysis.

7.1.16 Change in pitching moment coefficient due to elevator deflection

Also known as "elevator power" or "elevator effectiveness," this stability derivative is the change in pitching moment coefficient with a change in the elevator angle. The following equation calculates the derivative as the moment due to lift on the horizontal tail upon elevator deflection,

$$C_{m_{\delta_E}} = \frac{-l_t}{\bar{c}} C_{L_{\delta_E}} \tag{7-17}$$

The above derivatives constitute the non-dimensional, longitudinal stability derivatives.

7.2 Dimensional longitudinal stability derivatives

The respective dimensional, longitudinal stability derivatives are obtained through a series of equations outlined in (SC98, SM84). These derivatives are used directly in the formulation of the aircraft state-space problem.

7.2.1 Change in force in the x-direction with airspeed

The dimensional force in the x-direction due to changes in airspeed is given as:

$$X_u = \frac{\rho US}{m}\left(-\frac{U}{2}C_{D_u} - C_D\right) \tag{7-18}$$

For this analysis the variation in drag due to velocity denoted by the stability derivative, C_{D_u}, is assumed to be negligible.

7.2.2 Change in force in the z-direction with airspeed

The dimensional force in the z-direction due to changes in airspeed is given as:

$$Z_u = \frac{\rho US}{m}\left(-\frac{U}{2}C_{L_u} - C_L\right) \tag{7-19}$$

For this analysis the variation in lift due to velocity denoted by the stability derivative, C_{L_u}, is assumed to be negligible.

7.2.3 Change in pitching moment with airspeed

The dimensional pitching moment due to changes in airspeed is given as:

$$M_u = \frac{US\bar{c}}{I_{yy}}\left(\frac{U}{2}C_{m_u} + C_m\right) \tag{7-20}$$

For this analysis the variation in pitching moment due to velocity denoted by the stability derivative, C_{m_u}, is assumed to be negligible.

7.2.4 Change in force in the x-direction with angle of attack

The dimensional force in the x-direction due to varying angle of attack is given as:

$$X_\alpha = \frac{\rho U^2 S}{2m}\left(C_L - C_{D_\alpha}\right) \tag{7-21}$$

7.2.5 Change in force in the z-direction with angle of attack

The dimensional force in the z-direction due to varying angle of attack is given as:

$$Z_\alpha = -\frac{\rho U^2 S}{2m}\left(C_D + C_{L_\alpha}\right) \tag{7-22}$$

7.2.6 Change in pitching moment with angle of attack

The dimensional pitching moment due to varying angle of attack is given as:

$$M_\alpha = \frac{\rho U^2 S\bar{c}}{2I_{yy}}C_{m_\alpha} \tag{7-23}$$

7.2.7 Change in force in the x-direction with rate of change of angle of attack

The dimensional force in the x-direction due to varying rate of change of angle of attack is given as:

$$X_{\dot{\alpha}} = -\frac{\rho U S \bar{c}}{4m} C_{D_{\dot{\alpha}}} \tag{7-24}$$

7.2.8 Change in force in the z-direction with rate of change of angle of attack

The dimensional force in the z-direction due to varying rate of change of angle of attack is given as:

$$Z_{\dot{\alpha}} = -\frac{\rho U S \bar{c}}{4m} C_{L_{\dot{\alpha}}} \tag{7-25}$$

7.2.9 Change in pitching moment with rate of change of angle of attack

The dimensional pitching moment due to varying rate of change of angle of attack is given as:

$$M_{\dot{\alpha}} = \frac{\rho U S \bar{c}^2}{4I_{yy}} C_{m_{\dot{\alpha}}} \tag{7-26}$$

7.2.10 Change in force in the x-direction with pitch rate

The dimensional force in the x-direction due to varying pitch rate is given as:

$$X_q = -\frac{\rho U S \bar{c}}{4m} C_{D_q} \tag{7-27}$$

7.2.11 Change in force in the z-direction with pitch rate

The dimensional force in the z-direction due to varying pitch rate is given as:

$$Z_q = -\frac{\rho U S \bar{c}}{4m} C_{L_q} \tag{7-28}$$

7.2.12 Change in pitching moment with pitch rate

The dimensional pitching moment due to varying pitch rate is given as:

$$M_q = \frac{\rho U S \bar{c}^2}{4I_{yy}} C_{m_q} \tag{7-29}$$

7.2.13 Change in force in the x-direction with elevator deflection

The dimensional force in the x-direction due to a change in elevator deflection is given as:

$$X_{\delta_E} = -\frac{\rho U^2 S}{2m} C_{D_{\delta_E}} \tag{7-30}$$

7.2.14 Change in force in the z-direction with elevator deflection

The dimensional force in the z-direction due to a change in elevator deflection is given as:

$$Z_{\delta_E} = -\frac{\rho U^2 S}{2m} C_{L_{\delta_E}} \tag{7-31}$$

7.2.15 Change in pitching moment with elevator deflection

The dimensional pitching moment due to a change in elevator deflection is given as:

$$M_{\delta_E} = \frac{\rho U^2 S\bar{c}}{I_{yy}} C_{m_{\delta_E}} \tag{7-32}$$

This completes the list of dimensional longitudinal stability derivatives.

7.3 Non-dimensional lateral-directional stability derivatives

The next set of equations is used to determine the lateral-directional stability derivatives. These derivatives pertain to rotational motion (roll and yaw) about the x-body-axis and the z-body-axis and to translational motion in the y-body-axis. The rate derivatives, dependent on the roll rate, p, and yaw rate, r, are dimensionless.

7.3.1 Change in side-force coefficient with sideslip angle

Wing sweep and dihedral, the fuselage, and the vertical tail contribute to this stability derivative. Developing each contribution separately allows the total expression for this derivative in the form:

$$C_{y_\beta} = C_{y_{\beta_{wing}}} + C_{y_{\beta_{fuse}}} + C_{y_{\beta_{vert}}} \tag{7-33}$$

The wing contribution consists of the effects from wing sweep and wing dihedral. These contributions are given as:

$$C_{y_{\beta_{sweep}}} = C_L^2 \frac{6 \tan\Lambda \sin\Lambda}{AR(AR + 4\cos\Lambda)} \qquad \text{(TQ48)} \tag{7-34}$$

and

$$C_{y_{\beta_{dihedral}}} = -0.0001 \cdot |\Gamma| \cdot 57.3 \qquad \text{(HE68)} \qquad (7\text{-}35)$$

The fuselage contribution is given as:

$$C_{y_{\beta_{fuse}}} = -K_i C_{L_{\alpha_{fuse}}} \left(\frac{A_{bodyref}}{S} \right) \qquad \text{(HE68)} \qquad (7\text{-}36a)$$

In this equation, the wing-fuselage interference factor, K_i, depends on the location of the wing in relation to the body centerline. For wings positioned above the body centerline:

$$K_i = 1 - 0.85555 \frac{2z_w}{h} \qquad (7\text{-}36b)$$

For wings positioned below the body centerline the equation becomes:

$$K_i = 1 + 0.485 \frac{2z_w}{h} \qquad (7\text{-}36c)$$

The value for $C_{L_{\alpha_{fuse}}}$ is fixed depending on the general shape of the fuselage. For a round-shaped fuselage this is 0.0525 per radian and for a rectangular fuselage, 0.1243 per radian. The final contributor to C_{y_β} is the vertical tail. Its contribution is formulated as:

$$C_{y_{\beta_{vert}}} = -kC_{L_{\alpha_{vert}}} \left(0.724 + 3.06 \frac{S_v}{S(1+\cos\Lambda)} + 0.4 \frac{z_w}{h} + 0.009AR \right) \frac{S_v}{S} \qquad \text{(HE68)} \qquad (7\text{-}37)$$

In this equation, k is an empirical factor that depends upon the ratio of the vertical tail span to the fuselage diameter in the tail region (SM84). In the code this ratio is defined as:

$$PAR = \frac{b_v}{2r_1} \tag{7-38a}$$

For $PAR \le 2.0$, $k = 0.75$, while for $PAR \ge 3.5$, $k = 1.0$. Between these values, PAR is calculated using the linear equation:

$$k = 1 + 0.166(PAR - 3.5) \tag{7-38b}$$

The above equation for $C_{y_{\beta_{vert}}}$ introduces the lift-curve-slope of the vertical tail. (SD43) formulates this coefficient as:

$$C_{L_{\alpha_{vert}}} = 57.3\begin{pmatrix} 0.0000397694 AR_{eff_{vert}}^5 - 0.00069754 AR_{eff_{tail}}^4 + 0.00461464^3{}_{eff_{vert}} \\ - 0.158634 AR_{eff_{vert}}^2 + 0.0401979 AR_{eff_{vert}} + 0.00037328 \end{pmatrix} \tag{7-39}$$

7.3.2 Change in rolling moment coefficient with sideslip angle

This derivative is often called the "effective dihedral derivative." (PH49) formulates this derivative using contributions from wing dihedral, wing shape and position, and the vertical tail. This equation is given as:

$$C_{l_\beta} = C_{l_{\beta_{dih}}} + C_{l_{\beta_{wing}}} + C_{l_{\beta_{vert}}} + \Delta C_{l_{\beta_1}} + \Delta C_{l_{\beta_2}} \tag{7-40}$$

The wing dihedral contribution is

$$C_{l_{\beta_{dih}}} = 57.3 \cdot \Gamma \cdot \left(\begin{array}{l} \dfrac{-0.20833 - \sqrt{(0.0434 - 4.6296 \cdot (1.0052 - AR))}}{2.3148148} \\ + 0.7(1-\lambda)^2 \dfrac{(AR - 1.5)}{6.5} \end{array} \right) \cdot 0.0001 \tag{7-41}$$

The wing contribution is calculated assuming zero dihedral and is given by (QU56) to be

$$C_{l_{\beta_{wing}}} = C_L\left(-\frac{1.25\cdot(0.71\lambda+0.29)}{AR\lambda}+0.05\right) \qquad (7\text{-}42)$$

The vertical tail contribution arises from the normal force caused by sideslip and is given by the equation:

$$C_{l_{\beta_{vert}}} = -C_{l_{\alpha_{vert}}}\frac{S_v}{S}\frac{z_v}{b_w}\eta_v \qquad \text{(PH49)} \qquad (7\text{-}43)$$

The last two terms represent the interference effects between the wing-fuselage and the wing-vertical tail. These values are dependent upon the wing position and are summarized in the following table, which has been duplicated from (SM84).

TABLE 7.3.1: Values for interference effects on the vertical tail (1/rad)

	Wing-fuselage $\Delta C_{l_{\beta_1}}$	Wing-vertical Tail $\Delta C_{l_{\beta_2}}$
High Wing	-0.0006	0.00016
Mid Wing	0	0
Low Wing	0.0008	-0.00016

7.3.3 Change in yawing moment coefficient with sideslip angle

This derivative is often referred to as "weathercock stability" or as the static directional stability derivative. The method used to calculate this derivative includes contributions from the fuselage and the vertical tail and appears as:

$$C_{n_\beta} = -K_n\frac{S_{B_s}}{S}\frac{L_B}{b} - C_{y_{\beta_{vert}}}\frac{l_v}{b} \qquad \text{(HE68)} \qquad (7\text{-}42)$$

K_n is an empirical interference factor that is given as a function of aircraft geometry and Reynolds number (SM84).

7.3.4 Change in rolling moment coefficient with roll rate

This derivative is known as the roll-damping derivative. The main contribution comes from the wing however, the effects of the horizontal and vertical tails are also considered here. The complete formulation appears as:

$$C_{l_p} = C_{l_{pwing}} + C_{l_{Ptail}} + C_{l_{Pvert}} \tag{7-43}$$

In this equation the wing contribution is determined under the assumptions of zero or small sweep, small dihedral, and a linear lift curve slope. The resulting equation is:

$$C_{l_{pwing}} = C_{l_{paow}} - \frac{1}{8}\frac{C_L^2}{\pi AR\cos^2\Lambda}\left(1+2\sin^2\Lambda\frac{AR+2\cos\Lambda}{AR+4\cos\Lambda}\right) - \frac{1}{8}C_{D_0} \quad \text{(GA49)} \tag{7-44a}$$

where

$$C_{l_{paow}} = C_{l_{pao}}\left(\frac{AR+4\cos\Lambda}{2\frac{\pi AR}{C_{l_\alpha}}+4\cos\Lambda}\right) \quad \text{(BI49)} \tag{7-44b}$$

with

$$C_{l_{pao}} = 0.00000929805AR^5 - 0.00031818AR^4 + 0.00379426AR^3$$
$$- 0.0157796AR^2 - 0.0499261AR - 0.0500059 + 0.209\frac{(1-\lambda)^2(AR-1)}{9} \quad \text{(CM52)} \tag{7-44c}$$

The horizontal tail contribution is found in the same manner through the equation

$$C_{l_{Ptail}} = 0.5\frac{S_t}{S}\frac{b_t^2}{b^2}C_{l_{Paoh}}\left(\frac{AR_t + 4\cos\Lambda_t}{2\frac{\pi AR_t}{C_{l_{\alpha_{tail}}}} + 4\cos\Lambda_t}\right) \tag{7-45a}$$

with

$$C_{l_{Pao}} = 0.00000929805AR_t^5 - 0.00031818AR_t^4 + 0.00379426AR_t^3$$
$$-0.0157796AR_t^2 - 0.0499261AR_t - 0.0500059 + 0.209\frac{(1-\lambda_t)^2(AR_t - 1)}{9} \tag{7-45b}$$

The final contribution, that from the vertical tail, is given as:

$$C_{l_{Pvert}} = 2\left(\frac{z_v}{b}\right)^2 C_{y_{\beta_{vert}}} \tag{7-46}$$

7.3.5 Change in side-force coefficient with roll rate

The vertical tail is the main contributor to this derivative. It is formulated as:

$$C_{y_p} = \left(\frac{C_{y_p}}{C_L}\right)C_L + \left(\frac{\Delta C_{y_p}}{C_{l_{P\Gamma=0}}}\right)C_{l_p} \qquad \text{(HE68)} \tag{7-47b}$$

The ratios given in the above equation are determined by:

$$\left(\frac{C_{y_p}}{C_L}\right) = \frac{1}{10}\left[4 + \frac{7\cdot(57.3\Lambda - 30)}{60} - 3\cdot\left(1 - \lambda^{1.58495}\right)\right] \tag{7-47b}$$

and

$$\left(\frac{\Delta C_{y_p}}{C_{l_{P\Gamma=0}}}\right) = \frac{(\Gamma - 10)}{17.5} + 0.5 \tag{7-47c}$$

7.3.6 Change in yawing moment coefficient with roll rate

The wing and the vertical tail are the main contributors to this derivative. The form of the equation is simply:

$$C_{n_p} = C_{n_{p_{wing}}} + C_{n_{p_{vert}}} \tag{7-48}$$

The wing contribution is calculated as:

$$C_{n_{p_{wing}}} = C_L \frac{AR+4}{AR+4\cos\Lambda}\left[1+6\left(1+\frac{\cos\Lambda}{AR}\right)\frac{\tan^2\Lambda}{12}\right]\left(\frac{C_{n_p}}{C_L}\right)_{\Lambda=0^o} \qquad \text{(TQ48)} \tag{7-49a}$$

The ratio of C_{n_p} to C_L for unswept wings, as a function of AR, is found to be

$$\left(\frac{C_{n_p}}{C_L}\right)_{\Lambda=0^o} = -0.0000140468AR^3 + 0.000649352AR^2 - 0.0122523AR + 0.00192856 \tag{7-49b}$$

(WO51) presents the formulation of the vertical tail contribution as it is effected by variations in sidewash on the wing and fuselage. This relationship is given as:

$$C_{n_{p_{vert}}} = 57.3C_{l_{\alpha_{vert}}}\frac{S_v}{S}\frac{1}{b}\left(z_v\sin\alpha + l_v\cos\alpha\right)\left[\frac{2}{b}\left(z_v\cos\alpha - l_v\sin\alpha\right) - \left(\frac{\partial\sigma_1}{\partial\left(\frac{pb}{2U}\right)} + \frac{\partial\sigma_2}{\partial\left(\frac{pb}{2U}\right)}\right)\right] \tag{7-50}$$

Two terms are introduced in this relationship, which represent the effect of wing sidewash and fuselage sidewash. The first is given as a function of $2z_v/b$ or, as referred to within the analysis, P. The formulation from (SM84) follows:

$$\frac{\partial \sigma_1}{\partial\left(\frac{pb}{2U}\right)} = \sigma_2 + TR(\sigma_1 - \sigma_2) \tag{7-51a}$$

where

$$\sigma_1 = 22360.7P^9 - 69947.1P^8 + 724643P^7 - 70229.7P^6 + 32373.8P^5 - 9487.01P^4 + 1764.41P^3 - 200.25P^2 + 11.9843P - 0.00142786 \tag{7-51b}$$

and for $P \geq 0.157$

$$\sigma_2 = 0.533183P^2 - 0.78216P + 0.3517 \tag{7-51c}$$

otherwise,

$$\sigma_2 = 216915P^9 - 610122P^8 + 724643P^7 - 473060P^6 + 185140P^5 - 44463.6P^4 + 6444.43P^3 - 531.134P^2 + 21.1205P + 0.0395735 \tag{7-51d}$$

The fuselage sidewash effect, denoted by the second term, is given as:

$$\frac{\partial \sigma_2}{\partial\left(\frac{pb}{2U}\right)} = 9.30\left(\frac{\Delta h}{b}\right)^2 \frac{3}{b} \tag{7-52a}$$

where

$$\frac{\Delta h}{b} = \frac{z_v - z_v \cos\alpha - l_v \sin\alpha}{b} \tag{7-52b}$$

7.3.7 Change in side-force coefficient with yaw rate

The main contributions to this derivative come from the vertical tail and, to a small degree, the wing. The resulting equation is:

$$C_{y_r} = C_{y_{r_{wing}}} + C_{y_{r_{vert}}} \tag{7-53}$$

The corresponding components are given as:

$$C_{y_{r_{wing}}} = 0.143C_L - 0.05 \qquad \text{(GB48)} \qquad (7\text{-}54)$$

and

$$C_{y_{r_{vert}}} = -2\frac{l_v}{b}C_{y_{\beta_{vrt}}} \qquad \text{(CM52)} \qquad (7\text{-}55)$$

7.3.8 Change in rolling moment coefficient with yaw rate

The dominant contributors to this derivative are the wing and vertical tail. The basic relationship is:

$$C_{l_r} = C_{l_{r_{wing}}} + C_{l_{r_{vert}}} \qquad (7\text{-}56)$$

The wing contribution is calculated

$$C_{l_{r_{wing}}} = 0.05\frac{100Y}{100 - 57.3\Lambda}C_L \qquad \text{(HE68)} \qquad (7\text{-}57a)$$

where the dummy variable, Y, is given as

$$Y = -0.0000247674AR^4 + 0.00194154AR^3 - 0.0468052AR^2 + 0.456404AR - 0.0980299 + \frac{(AR+32.4)\left(\lambda^{1.12(1-\lambda)}\right)}{20.6511} + 2 \qquad \text{(SM84)} \qquad (7\text{-}57b)$$

The vertical tail contribution is simply:

$$C_{l_{r_{vert}}} = -2\frac{l_v z_v}{b^2}C_{y_{\beta_{tail}}} \qquad \text{(CG49)} \qquad (7\text{-}58)$$

7.3.9 Change in yawing moment coefficient with yaw rate

This derivative is known as the yaw damping derivative. The vertical tail is the main contributor though the following formulation also takes the effect of the wing into account. The sum of these effects yields:

$$C_{n_r} = C_{n_{r_{wing}}} + C_{n_{r_{vert}}} \tag{7-59}$$

In the above equation, the wing contribution is calculated as:

$$C_{n_{r_{wing}}} = -0.33\left(\frac{1+3\lambda}{2+2\lambda}\right)C_{D_0} - 0.02\left(1-\frac{AR-6}{13}-\frac{1-\lambda}{2.5}\right)C_L^2 \qquad \text{(CM43)} \qquad \text{(7-60)}$$

And the vertical tail contribution appears as:

$$C_{n_{r_{tail}}} = 2\left(\frac{l_v}{b}\right)^2 C_{y_{\beta_{tail}}} \qquad \text{(CM52)} \qquad \text{(7-61)}$$

7.3.10 Change in side-force coefficient with aileron deflection

This derivative is zero for most conventional, light aircraft and is assumed to be so throughout this analysis.

7.3.11 Change in rolling moment coefficient with aileron deflection

This derivative is commonly known as the aileron effectiveness or "aileron power." The basic equation is given as:

$$C_{l_{\delta_A}} = \left(\frac{C_{l_{\delta_A}}}{\tau}\right)\tau \qquad \text{(PJ38)} \qquad \text{(7-62)}$$

The method presented requires two steps to find the ratio of $C_{l_{\delta_A}}$ to τ. The first step is to calculate this ratio using the distance, P, from the body centerline to the outboard edge of the aileron divided by the wing semi-span. The following equations summarize the necessary calculations and restrictions from (SM84):

To calculate the required distance for the first step use the relation:

$$P = \frac{2Y_i}{b} \tag{7-63}$$

The desired ratio is dependent upon the aspect ratio and therefore is calculated three different ways. The first is for the case of $6 < AR < 10$:

$$\left(\frac{C_{l_{\delta_A}}}{\tau}\right) = \left(\frac{C_{l_{\delta_A}}}{\tau}\right)_B + \left[\left(\frac{C_{l_{\delta_A}}}{\tau}\right)_A - \left(\frac{C_{l_{\delta_A}}}{\tau}\right)_B\right]\frac{AR-6}{4} \tag{7-64a}$$

for $AR \le 6$:

$$\left(\frac{C_{l_{\delta_A}}}{\tau}\right) = \left(\frac{C_{l_{\delta_A}}}{\tau}\right)_B \tag{7-64b}$$

and for $AR \ge 10$

$$\left(\frac{C_{l_{\delta_A}}}{\tau}\right) = \left(\frac{C_{l_{\delta_A}}}{\tau}\right)_A \tag{7-64c}$$

These relationships refer to the following formulas and the corresponding ratios:

$$\left(\frac{C_{l_{\delta_A}}}{\tau}\right)_A = 2.0833P^5 - 6.1553P^4 + 5.36932P^3 - 0.567235P^2 + 0.170341P - 0.000227237 \tag{7-65}$$

and

$$\left(\frac{C_{l_{\delta_A}}}{\tau}\right)_B = 51.702P^8 - 238.294P^7 + 451.162P^6 - 444.985P^5 + 241.797P^4 - 70.2549P^3 + 10.5819P^2 - 0.408045P + 0.0000119225 \quad (7\text{-}66)$$

This completes the first step. The second step uses the distance from the body centerline to the inboard edge of the aileron divided by the wing semi-span:

$$P = \frac{2(Y_i + b_a)}{b} \quad (7\text{-}67)$$

Using this new value, the ratio of $C_{l_{\delta_A}}$ to τ is recalculated utilizing the same equations, under the same restrictions. The final value for this ratio is found as:

$$\left(\frac{C_{l_{\delta_A}}}{\tau}\right) = \left(\frac{C_{l_{\delta_A}}}{\tau}\right)_{Step2} - \left(\frac{C_{l_{\delta_A}}}{\tau}\right)_{Step1} \quad (7\text{-}68)$$

The final step in determining $C_{l_{\delta_A}}$ is to find τ, which is given as:

$$\tau = -65.1062\left(\frac{c_a}{c_{tail}}\right)^4 + 50.8227\left(\frac{c_a}{c_{tail}}\right)^3 - 15.7949\left(\frac{c_a}{c_{tail}}\right)^2 + 3.53383\left(\frac{c_a}{c_{tail}}\right) + 0.000043467 \quad (7\text{-}69)$$

7.3.12 Change in yawing moment coefficient with aileron deflection

The equation for this derivative is given as:

$$C_{n_{\delta_A}} = 2KC_L C_{l_{\delta_A}} \quad \text{(HE68)} \quad (7\text{-}70)$$

The empirical factor, K, is dependent on the wing aspect ratio and the distance from the body centerline to the inboard edge of the aileron divided by the wing semi-span. This distance appears as:

$$R = \frac{2Y_i}{b} \tag{7-71}$$

For all values of the wing aspect ratio, K is calculated as:

$$K = e_{f_2} + \frac{\lambda - 0.25}{0.75}\left(e_{f_1} - e_{f_2}\right) \tag{7-72}$$

The following series of equations and relationships presents the method for determining the terms e_{f_1} and e_{f_2} from (SM84):

For $AR < 3$:

$$e_{f_1} = -0.36 \tag{7-73a}$$

$$e_{f_2} = -0.0889976R^3 - 0.00666109R^2 + 0.0419053R - 0.284843 \tag{7-73b}$$

For $3 \le AR \le 4$:

$$e_{f_1} = A_{f_1} + (AR - 3)\left(B_{f_1} - A_{f_1}\right) \tag{7-73c}$$

$$A_{f_1} = -0.36 \tag{7-73d}$$

$$B_{f_1} = 0.0538037R^3 - 0.133855R^2 + 0.00627854R - 0.262321 \tag{7-73e}$$

and

$$e_{f_2} = A_{f_2} + (AR - 3)\left(B_{f_2} - A_{f_2}\right) \tag{7-73f}$$

$$A_{f_2} = -0.0889976R^3 - 0.00666109R^2 + 0.0419053R - 0.28483 \quad (7\text{-}73g)$$

$$B_{f_1} = 0.0549387R^3 - 0.164298R^2 + 0.101864R - 0.234861 \quad (7\text{-}73h)$$

For $4 < AR < 6$:

$$e_{f_1} = A_{f_1} + 0.5(AR - 4)(B_{f_1} - A_{f_1}) \quad (7\text{-}73i)$$

$$A_{f_1} = 0.0538037R^3 - 0.133855R^2 + 0.00627854R - 0.262321 \quad (7\text{-}73j)$$

$$B_{f_1} = -0.0629823R^3 - 0.0173375R^2 - 0.0333427R - 0.180026 \quad (7\text{-}73k)$$

and

$$e_{f_2} = A_{f_2} + 0.5(AR - 4)(B_{f_2} - A_{f_2}) \quad (7\text{-}73l)$$

$$A_{f_2} = 0.0549387R^3 - 0.164298R^2 + 0.101864R - 0.234861 \quad (7\text{-}73m)$$

$$B_{f_2} = -0.108911R^3 + 0.0291149R^2 + 0.0440961R - 0.160312 \quad (7\text{-}73n)$$

For $6 \leq AR \leq 8$:

$$e_{f_1} = A_{f_1} + 0.5(AR - 6)(B_{f_1} - A_{f_1}) \quad (7\text{-}73o)$$

$$A_{f_1} = -0.0629823R^3 - 0.0173375R^2 - 0.0333427R - 0.180026 \quad (7\text{-}73p)$$

$$B_{f_1} = -0.108911R^3 + 0.0291149R^2 + 0.0440961R - 0.160312 \quad (7\text{-}73q)$$

and

$$e_{f_2} = A_{f_2} + 0.5(AR - 6)(B_{f_2} - A_{f_2}) \quad (7\text{-}73r)$$

$$A_{f_2} = -0.108911R^3 + 0.0291149R^2 + 0.0440961R - 0.160312 \quad (7\text{-}73s)$$

$$B_{f_2} = -0.191365R^3 + 0.147284R^2 - 0.00039296R - 0.119884 \quad (7\text{-}73t)$$

and finally, for $AR > 8$:

$$e_{f_1} = -0.110526R^2 + 0.013893R - 0.146355 \qquad \text{(7-73u)}$$

$$e_{f_2} = -0.191365R^3 + 0.147284R^2 - 0.00039296R - 0.119884 \qquad \text{(7-73v)}$$

7.3.13 Change in side-force coefficient with rudder deflection

(ET59) gives the following formula for estimating this derivative:

$$C_{y_{\delta_R}} = C_{l_{\alpha_{vert}}} \tau \frac{S_v}{S} \qquad \text{(7-74a)}$$

where

$$\tau = 21.7949\left(\frac{S_r}{S_v}\right)^5 - 46.4744\left(\frac{S_r}{S_v}\right)^4 + 36.9347\left(\frac{S_r}{S_v}\right)^3 - 14.259\left(\frac{S_r}{S_v}\right)^2 + 3.70551\left(\frac{S_r}{S_v}\right) - 0.000057815 \qquad \text{(7-74b)}$$

7.3.14 Change in rolling moment coefficient with rudder deflection

The equation for calculating this derivative is:

$$C_{l_{\delta_R}} = C_{l_{\alpha_{vert}}} \tau \frac{S_v}{S} \frac{z_v}{b} \qquad \text{(ET59)} \qquad \text{(7-75)}$$

The value of τ is calculated by using the same equation given in the development of $C_{y_{\delta_R}}$.

7.3.15 Change in yawing moment coefficient with rudder deflection

This derivative is known as the "rudder power." (PH49) gives the following formulation for this derivative:

$$C_{n_{\delta_R}} = -C_{l_{\alpha_{vert}}} \tau \frac{S_v}{S} \frac{l_v}{b} \eta_v \tag{7-76}$$

The value of τ is calculated by using the same equation given in the development of $C_{y_{\delta_R}}$.

7.4 Dimensional lateral-directional stability derivatives

The respective dimensional, lateral-directional stability derivatives are obtained through a series of equations outlined in (SC98, SM84). These derivatives are used directly in the formulation of the aircraft state-space problem.

7.4.1 Change in force in the y-direction with sideslip angle

The dimensional force in the y-direction due to changes in sideslip angle is given as:

$$Y_\beta = \frac{\rho U^2 S}{2m} C_{y_\beta} \tag{7-77}$$

7.4.2 Change in rolling moment with sideslip angle

The dimensional rolling moment due to changes in sideslip angle is:

$$L_\beta = \frac{\rho U^2 S}{2I_{xx}} C_{l_\beta} \tag{7-78}$$

7.4.3 Change in yawing moment with sideslip angle

The dimensional yawing moment due to changes in sideslip angle is:

$$N_\beta = \frac{\rho U^2 Sb}{2I_{zz}} C_{n_\beta} \tag{7-79}$$

7.4.4 Change in force in the y-direction with roll rate

The dimensional force in the y-direction due to variations in roll rate is:

$$Y_p = \frac{\rho USb}{4m} C_{y_p} \tag{7-80}$$

7.4.5 Change in rolling moment with roll rate

The dimensional rolling moment due to variations in roll rate is:

$$L_p = \frac{\rho USb^2}{4I_{xx}} C_{l_p} \tag{7-81}$$

7.4.6 Change in yawing moment with roll rate

The dimensional yawing moment due to variations in roll rate is:

$$N_p = \frac{\rho USb^2}{4I_{zz}} C_{n_p} \tag{7-82}$$

7.4.7 Change in force in the y-direction with yaw rate

The dimensional force in the y-direction due to variations in yaw rate is:

$$Y_r = \frac{\rho USb}{4m} C_{y_r} \tag{7-83}$$

7.4.8 Change in rolling moment with yaw rate

The dimensional rolling moment due to variations in yaw rate is:

$$L_r = \frac{\rho USb^2}{4I_{xx}} C_{l_r} \tag{7-84}$$

7.4.9 Change in yawing moment with yaw rate

The dimensional yawing moment due to variations in yaw rate is:

$$N_r = \frac{\rho USb^2}{4I_{zz}} C_{n_r} \tag{7-85}$$

7.4.10 Change in force in the y-direction with aileron deflection

The dimensional force in the y-direction due to changing aileron deflection is:

$$Y_{\delta_A} = \frac{\rho U^2 S}{2m} C_{y_{\delta_A}} \tag{7-86}$$

7.4.11 Change in rolling moment with aileron deflection

The dimensional rolling moment due to changing aileron deflection is:

$$L_{\delta_A} = \frac{\rho U^2 Sb}{2I_{xx}} C_{l_{\delta_A}} \tag{7-87}$$

7.4.12 Change in yawing moment with aileron deflection

The dimensional yawing moment due to changing aileron deflection is:

$$N_{\delta_A} = \frac{\rho U^2 Sb}{2I_{zz}} C_{n_{\delta_A}} \tag{7-88}$$

7.4.13 Change in force in the y-direction with rudder deflection

The dimensional force in the y-direction due to changing rudder deflection is:

$$Y_{\delta_R} = \frac{\rho U^2 S}{2m} C_{y_{\delta_R}} \tag{7-89}$$

7.4.11 Change in rolling moment with rudder deflection

The dimensional rolling moment due to changing rudder deflection is:

$$L_{\delta_R} = \frac{\rho U^2 Sb}{2I_{xx}} C_{l_{\delta_R}} \tag{7-90}$$

7.4.12 Change in yawing moment with rudder deflection

The dimensional yawing moment due to changing rudder deflection is:

$$N_{\delta_R} = \frac{\rho U^2 Sb}{2I_{zz}} C_{n_{\delta_R}} \tag{7-91}$$

This completes the list of dimensional lateral-directional stability derivatives.

8. STATIC STABILITY AND CONTROL

The S&C derivatives presented in the previous chapter are now used for static stability and control analysis. This chapter presents static S&C analysis techniques used to guarantee desirable S&C characteristics, as will be defined in the set of constraints used to drive the design optimization process.

8.1 Ground handling stability

The ground handling stability of an aircraft refers to stability on the ground during taxiing and takeoff ground roll. The important factors laid out by (RO185) are the tip-over criterion, the longitudinal ground clearance criterion, and the loading distribution on the landing gear. Tricycle landing gear is assumed to be standard for the type of aircraft considered by this analysis.

8.1.1 Tip-over criterion

The tip-over criterion is based on statistical data, which shows that the angle made between the main landing gear and the center of gravity location is typically 15 degrees. Based on the aircraft geometry:

$$tipangle = \tan^{-1}\left(\frac{X_{gear}}{Z_{gear}}\right) \tag{8-1a}$$

where

$$X_{gear} = CG_{mgr} - XM \tag{8-1b}$$

$$Z_{gear} = 0.5h_{gear} + z_t + 0.5l_m \tag{8-1c}$$

8.1.2 Longitudinal ground clearance criterion

This ground clearance criterion requires that the aircraft have the capability to rotate to the angle required for lift-off without hitting the tail of the aircraft on the runway. Statistical data shows that this is typically 15 degrees but may be more or less depending on the incidence angle of the wing and the geometric angle of attack due to the landing gear. As a conservative estimate, the wing incidence angle is not accounted for and the following approximation is used:

$$clearance = \tan^{-1}\left(\frac{Z_{gear}}{L_B - CG_{mgr}}\right) \tag{8-2}$$

8.1.3 Maximum static load per strut

This requirement on the maximum static load per strut ensures stability on the ground as well as the ability to rotate for takeoff. Excessive loading either way creates problems in both arenas. The formulation taken from (RO185) is:

$$P_{nose} = \left(\frac{W_{guess} X_{gear}}{X_{gear} + XM - CG_{ngr}}\right) \tag{8-3}$$

$$P_{main} = \left(\frac{W_{guess}\left(XM - CG_{ngr}\right)}{N_{strut}\left(X_{gear} + XM - CG_{ngr}\right)}\right) \tag{8-4}$$

8.2 Control effectiveness and travel

The surface area of the elevator, rudder, and ailerons must be such that the control deflection required to maintain or alter a given condition does not exceed the physical limitations on the maximum and minimum travel. This restriction dictates the control effectiveness and directly affects the sizing of control surfaces.

8.2.1 Static elevator control

In this case static elevator control refers to the ability to obtain the maximum lift coefficient in steady, level, unaccelerated flight. At this condition, the lift and moment coefficients can be expressed as:

$$C_{L_0} + C_{L_\alpha}\alpha + C_{L_{\delta_E}}\delta_E = C_{L_{\max}} \tag{8-5a}$$

$$C_{m_0} + C_{m_\alpha}\alpha + C_{m_{\delta_E}}\delta_E = 0 \tag{8-5b}$$

These relationships yield the following result for the required elevator deflection:

$$\begin{bmatrix} \alpha \\ \delta_E \end{bmatrix} = \begin{bmatrix} C_{L_\alpha} & C_{L_{\delta_E}} \\ C_{m_\alpha} & C_{m_{\delta_E}} \end{bmatrix}^{-1} \begin{bmatrix} C_{L_{\max}} - C_{L_0} \\ -C_{m_0} \end{bmatrix} \tag{8-6}$$

8.2.2 Rotation

Another requirement on the elevator is the ability to induce rotation at takeoff. This control requirement can be derived by summing the moments about either the aircraft center of gravity or point of contact of the main landing gear with the runway. As a

rough approximation, the thrust and drag forces are assumed to be equal and therefore do not appear in the following equation:

$$\delta_E = \frac{\left[\frac{W\left(CG_{mgr} - XM\right)}{.5\rho U_{rotate}^2 S\bar{c}} - C_{m_0} - C_{m_\alpha}\alpha_{ground} - \left(\frac{CG_{mgr} - XM}{\bar{c}}\right)C_{L_0} - \left(\frac{CG_{mgr} - XM}{\bar{c}}\right)C_{L_\alpha}\alpha_{ground}\right]}{C_{m_{\delta_E}} + \left(\frac{CG_{mgr} - XM}{\bar{c}}\right)C_{L_{\delta_E}}} \tag{8-7a}$$

where

$$U_{rotate} = 1.11U_{takeoff} \qquad \text{(RO185)} \tag{8-7b}$$

and

$$\alpha_{ground} = \tan^{-1}\left(\frac{1/12\left(l_n - l_m\right)}{CG_{mgr} - CG_{ngr}}\right) + i_{wing} \tag{8-7c}$$

8.2.3 Static roll rate

This requirement imposes a condition on the required aileron deflection based upon a user defined static roll rate. In order to achieve this roll rate, the maximum/minimum aileron deflection must allow for:

$$\delta_a = \frac{-p_{static}L_p}{L_{\delta_a}} \qquad \text{(SC98)} \tag{8-8}$$

8.2.4 Control in sideslip

This requirement effects both the rudder and the ailerons, which must be implemented to maintain stable flight under the influence of a crosswind. This crosswind is approximated by the effective sideslip angle and is usually set at +/- 10 degrees (SC98). Solving the following relationship (SC98) provides the required rudder and aileron deflections needed for a user-specified sideslip angle:

$$\begin{Bmatrix} \Delta\phi \\ \Delta\delta_r \\ \Delta\delta_a \end{Bmatrix} = -\begin{bmatrix} C_L & C_{y_{\delta_r}} & C_{y_{\delta_a}} \\ 0 & C_{l_{\delta_r}} & C_{l_{\delta_a}} \\ 0 & C_{n_{\delta_r}} & C_{n_{\delta_a}} \end{bmatrix}^{-1} \begin{Bmatrix} C_{y_\beta} \\ C_{l_\beta} \\ C_{n_\beta} \end{Bmatrix} \Delta\beta \qquad \text{(SC98)} \qquad (8\text{-}9)$$

9. DYNAMIC STABILITY AND CONTROL ANALYSIS

The ability to perform dynamic stability and control analysis during the early design phase is rarely available. However, combining the computationally-fast applied aerodynamic analysis presented in the previous chapters, with stability derivatives and the analysis presented in this chapter leads to a computationally-efficient dynamic S&C analysis capability that is suitable for design optimization studies. The analysis methods presented for both the longitudinal and lateral-directional cases are based on (SC98).

9.1 Longitudinal Dynamics

Longitudinal dynamics refers to the aircraft's behavior about the y-body in the plane of symmetry.

9.1.1 Linearized equations of motion

The linearized equations of motion governing this system are (SC98):

$$U(\dot{u}/U) = UX_u(u/U) + X_\alpha\alpha - g\cos\Theta_0\theta + X_\delta\delta \tag{9-1a}$$

$$(U - Z_{\dot{\alpha}})\dot{\alpha} = UZ_u(u/U) + Z_\alpha\alpha + (U + Z_q)q - g\sin\Theta_0\theta + Z_\delta\delta \tag{9-1b}$$

$$-M_{\dot{\alpha}}\dot{\alpha} + \dot{q} = UM_u(u/U) + M_\alpha\alpha + M_q q + M_\delta\delta \tag{9-1c}$$

These equations can be written in the form of a linear time-invariant state space problem:

$$\{\dot{x}\} = [A]\{x\} + \{B\}\delta_E \tag{9-2}$$

where the 4 x 4 longitudinal airframe plant matrix, $[A]$, is given by:

(9-3)

$$A_{11} = X_u \qquad A_{12} = X_\alpha / U \qquad A_{13} = 0 \qquad A_{14} = -(g/U)\cos\theta$$

$$A_{21} = UZ_u/(U - Z_{\dot{\alpha}}) \qquad A_{22} = Z_\alpha/(U - Z_{\dot{\alpha}})$$

$$A_{23} = (U + Z_q)/(U - Z_{\dot{\alpha}}) \qquad A_{24} = -g\sin\theta/(U - Z_{\dot{\alpha}})$$

$$A_{31} = UM_u + M_{\dot{\alpha}}UZ_u/(U - Z_{\dot{\alpha}}) \qquad A_{32} = M_\alpha + M_{\dot{\alpha}}Z_\alpha/(U - Z_{\dot{\alpha}})$$

$$A_{33} = M_q + M_{\dot{\alpha}}(U + Z_q)/(U - Z_{\dot{\alpha}}) \qquad A_{34} = 0$$

$$A_{41} = A_{42} = A_{44} = 0 \qquad A_{43} = 1$$

the 4 x 1 longitudinal airframe control vector is:

(9-4)

$$B_1 = X_{\delta_E}/U \qquad B_2 = Z_{\delta_E}/(U - Z_{\dot{\alpha}})$$

$$B_3 = M_{\delta_E} + M_{\dot{\alpha}}Z_{\delta_E}/(U - Z_{\dot{\alpha}}) \qquad B_4 = 0$$

and the longitudinal state vector is:

(9-5)

$$\{x\} = [u/U \quad \alpha \quad q \quad \theta]^T$$

The task of solving the homogeneous form of the above state-space system of equations for the longitudinal modes is now a standard eigenvalue problem.

9.1.2 Eigenvalue interpretation

In the above analysis, the eigenvalues of the plant matrix, [*A*], are expected (in most conventional airplane designs) to be two conjugate pairs of roots corresponding to the short period and long-period (phugoid) modes of motion. This calculation is performed using the *eig*[] function in *MATLAB* (UM99). From these roots the modal damping ratio, natural frequency, and damped frequency are easily determined using the *damp*[] function in *MATLAB*. These parameters provide a measure of dynamic stability in motion about the pitch axis.

9.1.3 Modal approximations

For the purposes of optimization, a simplified method of approximating the modes of motion is also included. This is desirable to prevent unexpected eigenvalues (out of the 4 x 4 eigenvalue problem) from failing the optimization, if in the course of airplane variations any longitudinal poles become real and not complex conjugate as assumed a-priori. The short period approximation employed is:

$$\omega_n = \left(-M_\alpha\right)^{1/2} \tag{9-6}$$

$$\zeta = -\left(M_{\dot{\alpha}} + M_q\right)/\left(2\omega_n\right) \tag{9-7}$$

The phugoid approximation is:

$$\omega_n = \sqrt{2}g/U \tag{9-8}$$

$$\zeta = C_D \Big/ \sqrt{2}C_L \tag{9-9}$$

9.1.4 Control response

The dynamic response of the aircraft due to elevator input can now be investigated by incorporating the control vector, [*B*]. There are a number of direct methods to perform these calculations and *MATLAB* already possesses two functions in particular which address this problem. These are the *step*[] and *impulse*[] functions. The *step*[] function returns the time-domain control response to an elevator step input. The *impulse*[] function returns the time-domain control response to an elevator impulse input. In both cases, the non-dimensional speed, angle-of-attack, pitch rate, and pitch angle are calculated and plotted over time.

9.2 Lateral-directional dynamics

Lateral-directional dynamics refers to the aircraft's behavior about the yaw and roll axes.

9.2.1 Linearized equations of motion

The linearized equations of motion governing the lateral-directional system are (SC98):

$$U\dot{\beta} = Y_\beta \beta + Y_p p + g\cos\Theta_0 \phi + (Y_r - U)r + Y_\delta \delta \tag{9-10a}$$

$$\dot{p} - (I_{xz}/I_{xx})\dot{r} = L_\beta \beta + L_p p + L_r r + L_\delta \delta \tag{9-10b}$$

$$-(I_{xz}/I_{zz})\dot{p} + \dot{r} = N_\beta \beta + N_p p + N_r r + N_\delta \delta \tag{9-10c}$$

$$\dot{\phi} = p \tag{9-10d}$$

As before, these are written in the form of linear time-invariant state space equations as:

$$\{\dot{x}\} = [A]\{x\} + \{B\}\delta_{R,A} \tag{9-11}$$

where the 4 x 4 longitudinal airframe plant matrix, $[A]$, is:

(9-12)

$$A_{11} = Y_\beta / U \qquad A_{12} = Y_p / U \qquad A_{13} = (g/U)\cos\theta \qquad A_{14} = (Y_r - U)/U$$

$$A_{21} = G\left[L_\beta + N_\beta (I_{xz}/I_x)\right] \qquad A_{22} = G\left[L_p + N_p (I_{xz}/I_x)\right]$$

$$A_{23} = 0 \qquad A_{24} = G\left[L_r + N_r (I_{xz}/I_x)\right]$$

$$A_{31} = A_{33} = A_{34} = 0 \qquad A_{32} = 1$$

$$A_{41} = G\left[N_\beta + L_\beta (I_{xz}/I_z)\right] \qquad A_{42} = G\left[N_p + L_p (I_{xz}/I_z)\right]$$

$$A_{43} = 0 \qquad A_{44} = G\left[N_r + L_r (I_{xz}/I_z)\right]$$

and the 4 x 1 longitudinal airframe control vector is:

(9-13)

$$B_1 = Y_\delta / V \qquad B_2 = G\left[L_\delta + N_\delta (I_{xz}/I_x)\right]$$

$$B_3 = 0 \qquad B_4 = G\left[N_\delta + L_\delta (I_{xz}/I_z)\right]$$

and the longitudinal state vector is:

(9-14)

$$\{x\} = [\beta \quad p \quad \phi \quad r]^T$$

The constant G is defined as

$$G = \frac{I_x I_z}{I_x I_z - I_{xz}^2} \tag{9-15}$$

9.2.2 Eigenvalue interpretation

For the lateral-directional problem, the expected eigenvalues (in the case of most airplane designs) consist of one conjugate pair along with two real roots. These correspond to the Dutch-roll, roll and spiral response modes, respectively. These roots, along with the dutch roll damping ratio, natural frequency, and damped frequency are determined using the same *MATLAB* functions used in the longitudinal case. For the roll mode the time constant must be determined separately. This relationship is:

$$\tau_{roll} = \frac{-1}{\lambda_{roll}} \tag{9-16}$$

9.2.3 Modal approximations

The following approximation is implemented to protect against problems arising from possible disappearance in the course of optimization searches of the Dutch-roll complex-conjugate roots . This Dutch-roll approximation is written (SC98):

$$\omega_n = \sqrt{N_\beta} \tag{9-17}$$

$$\zeta = -0.5\frac{N_r}{\sqrt{N_\beta}} \tag{9-18}$$

9.2.4 Control response

The dynamic response of the aircraft due to aileron or rudder inputs is determined by incorporating the control vector, [*B*]. The *MATLAB* functions used in the longitudinal

case are used here as well. For impulse and step inputs of both surfaces, the sideslip angle, roll rate, bank angle, and yaw rate are calculated and can be plotted over time.

10. DESIGN OPTIMIZATION

While a major facet to this work was the development of the design-oriented computationally-fast preliminary design analysis code, the primary interest was to combine this capability with a numerical optimizer to create a design synthesis capability. This chapter presents the integrated optimization problem (specifically, the list of constraints accounted for) and describes integration with the MATLAB (UM99) numerical optimizer.

10.1 The Multivariable Constrained Nonlinear Optimization Problem

The general nonlinear optimization problem can be written simply as (VA99):

Minimize:	$F(X)$		objective function	(10-1)
Subject to:				
	$C_j(X) \le 0$	$j = 1, m$	inequality constraints	(10-2)
	$Ceq_k(X) = 0$	$k = 1, l$	equality constraints	(10-3)
	$X_i^l \le X_i \le X_i^u$	$i = 1, n$	side constraints	(10-4)

where X is the set of design variables used for optimization. The objective function, $F(X)$, is that parameter which is to be minimized (i.e. aircraft weight). The inequality constraints place restrictions on the absolute or relative values of different behavior measures and failure modes obtained through analysis by establishing limits of

acceptability and margins of safety. The equality constraints set limitations that force the specified values of behavior measures or combinations of design variables to match exactly. The side constraints are merely lower and upper bounds on the design variables themselves.

10.2 Optimizer selection and integration

The *FMINCON* function from *MATLAB* was selected to perform the task of optimization for this research. It is a standard nonlinear programming optimizer based on quadratic programming. A part of the *MATLAB Optimization Toolbox*, this function finds the constrained minimum of a function of several variables. Any other robust nonlinear programming optimizer can be easily integrated with the present analysis code. The present choice was driven by its straightforward integration with the MATLAB analysis modules and its good performance. By creating three new files to comply with *FMINCON* formatting, the linking of the analysis code and the optimizer was completed. These files and their functions are described in the following table:

TABLE 10.2.1 Optimization Files

NAME	FUNCTION
OPT1st.m	Main executable. Loads user inputs and launches the optimizer.
ANALfun1st.m	Function to call analysis code. Called by *FMINCON*.
NONLCON.m	Function to return constraint values. Called by *FMINCON*.

10.3 Flow of Information

The following figure presents the flow of information throughout the optimization process:

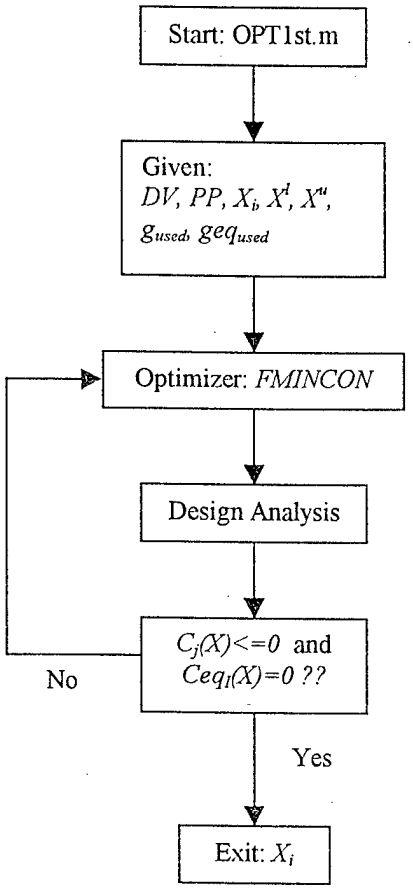


Figure 10.1. Basic Optimization Flow Diagram

Execution starts with OPT1st.m, where the code begins and the set of all required user inputs are read and forwarded to the optimizer. These inputs are detailed in Chapter 2. Once the variables and constraints to be used are sent to the optimizer, control of execution is assumed by the optimizer. It will call the analysis program to establish a

baseline set of values for the constraint vector based on the initial values for the design variables. On subsequent iterations it will converge on a vector of design variables which satisfies all constraints and renders the objective function optimal. In the course of the search for the optimum the optimizer varies the design variable vector numerous times, and sends it to the analysis code. The analysis code, in turn, returns values of all constraints and of the objective function to the optimizer. Once convergence has been accomplished, the final design variable vector is returned and the optimization is complete.

10.4 Constraint Definition

For optimization purposes, the user is presented with a minimum of 53 inequality constraints and 5 equality constraints from which to formulate various problems. A complete list of these constraints may be found in Table 10.2.

As explained in Appendix A, the files, *ConstraintsUsed.wk1* and *EQconsUsed.wk1*, allow these constraints to be used or disregarded based on the user's inputs. The constraints are grouped as follows:

TABLE 10.4.1: Geometric Constraints

NUMBER	DESCRIPTION
g(1)	Inboard edge of the aileron plus the aileron span must be less than the wing semispan.
g(2)	The inboard edge of the flap plus the flap span must be less than the distance to the inboard edge of the aileron.
g(3)	The horizontal tail span must be less than the wing span.
g(4)	The vertical tail span must be less than the wing span.
g(5)	The individual elevator span must be less than the semispan of the horizontal tail.
g(6)	The rudder span must be less than the span of the vertical tail.
g(7)	The aileron chord must be less than 20% of the wing chord.
g(8)	The elevator chord must be less than half the horizontal tail chord.
g(9)	The rudder chord must be less than half the vertical tail chord.
g(10)	The horizontal tail must be located on the fuselage.
g(11)	The vertical tail must be located on the fuselage.
g(12)	The leading edge of the horizontal tail must be behind the trailing edge of the wing.
g(13)	The leading edge of the vertical tail must be behind the trailing edge of the wing.
g(14)	The center of gravity must be located on the fuselage.
g(15)	The main gear must be located aft of the nose gear.
g(16)	The center of gravity must be located aft of the nose gear.
g(17)	The center of gravity must be located ahead of the main gear.
g(18)	The effective aspect ratio of the vertical tail must be less than 6.5 due to limitations on stability and control equations.
g(19)	The effective aspect ratio of the vertical tail must not be negative.
g(20)	The aspect ratio of the horizontal tail must be less than 10 due to limitations on stability and control equations.
g(21)	The aspect ratio of the horizontal tail must be greater than zero.
g(22)	The aspect ratio of the wing must be less than 8 due to limitations on stability and control equations.
g(23)	The aspect ratio of the wing must be greater than 4 due to limitations on stability and control equations.
g(24)	The static margin must be less than the max allowable static margin.
g(25)	The static margin must be greater than the min allowable static margin.

TABLE 10.4.2: Static Stability Constraints

NUMBER	DESCRIPTION
g(26)	Require C_{y_β} to be negative for lateral-directional stability.
g(27)	Require C_{l_β} to be negative for lateral-directional stability.
g(28)	Require C_{n_β} to be positive for lateral-directional stability.
g(29)	Require C_{y_β} to be greater than the user defined minimum value.
g(30)	Require C_{l_β} to be greater than the user defined minimum value.
g(31)	Require C_{n_β} to be less than the user defined maximum value.

TABLE 10.4.3: Control Surface Deflection Constraints

NUMBER	DESCRIPTION
g(32)	Required elevator for static control must be less than defined max.
g(33)	Required elevator for static control must be greater than defined min.
g(34)	Required elevator for rotation must be greater than defined min.
g(35)	Required aileron for static control must be greater than defined min.
g(36)	Required aileron for static control must be less than defined max.
g(37)	Rudder required in a $+\beta$ crosswind must be less than defined max.
g(38)	Rudder required in a $+\beta$ crosswind must be greater than defined min.
g(39)	Rudder required in a $-\beta$ crosswind must be less than defined max.
g(40)	Rudder required in a $-\beta$ crosswind must be greater than defined min.
g(41)	Aileron required in a $+\beta$ crosswind must be less than defined max.
g(42)	Aileron required in a $+\beta$ crosswind must be greater than defined min.
g(43)	Aileron required in a $-\beta$ crosswind must be less than defined max.
g(44)	Aileron required in a $-\beta$ crosswind must be greater than defined min.

TABLE 10.4.4: Dynamic Stability Constraints

NUMBER	DESCRIPTION
g(45)	The phugoid damping ratio must be greater than defined min.
g(46)	The short period damping ratio must be greater than defined min.
g(47)	The short period damping ratio must be less than defined max.
g(48)	The short period natural frequency must be less than defined max.
g(49)	The short period natural frequency must be less than defined min.
g(50)	The Dutch roll damping ratio must be greater than defined min.
g(51)	The roll mode time constant must be less than defined max.

TABLE 10.4.5: Fuel Weight Constraints

NUMBER	DESCRIPTION
g(52)	The amount of fuel needed for the mission must be less than the maximum fuel capacity.
g(53)	The amount of fuel needed must not be negative.

The following mission constraints are added to the end of the *g*-vector constraints given above. Thus, the number of constraints varies depending on the mission profile.

TABLE 10.4.6: Mission Performance Constraints

NUMBER	DESCRIPTION
gTO(1)	The balanced field length for takeoff must be less than the runway length.
gTO(2)	The minimum takeoff speed must be less than or equal to the desired takeoff speed.
gTO(3)	The power required for takeoff must be less than or equal to the power available.
gTO(4)	The rate of climb at takeoff must be greater than or equal to desired rate of climb.
gCR(1)	The rate of climb at cruise must be greater than or equal to the desired rate of climb.

TABLE 10.4.6: Mission Performance Constraints (continued)

gCR(2)	The power available at cruise must be greater than or equal to the power required.
gCR(3)	The cruise Mach number must be less than upper Mach limit.
gCR(4)	The maximum possible speed must be greater than the max wind speed.
gLT(1)	The power available at loiter must be greater than or equal to the power required.
gLT(2)	The loiter Mach number must be less than upper Mach limit.
gLT(3)	The maximum possible speed must be greater than the max wind speed.
gLD(1)	The required runway length for landing must be less than the actual runway length.
gLD(2)	The landing speed must be less than the specified landing speed.

TABLE 10.4.7: Equality Constraints

The equality constraints are formulated in the *geq*-vector, separate from the inequality constraints. These address issues of ground handling and stability as well as gross weight convergence.

NUMBER	DESCRIPTION
geq(1)	The ground handling tip-angle should be approximately 15 degrees.
geq(2)	The longitudinal ground-clearance angle should be approximately 15 degrees.
geq(3)	The load on the nose gear should be 10% of aircraft weight.
geq(4)	The combined load on the main gear should be 90% of aircraft weight.
geq(5)	The calculated weight must converge to the guessed weight.

The constraints are formulated within the Allconstraints.m module, as indicated in Appendix A. Within this module each constraint has been normalized so that the order of each entry is unity. This technique is recommended whenever formulating a problem such as this, which involves many multidisciplinary constraints with different relative

magnitudes, as it improves the conditioning of the optimization problem considerably (VA99). One example might be to require the power required, for a particular mission leg, be less than the power available in the form of an inequality constraint:

$$P_{required} < P_{available} \tag{10-5}$$

and in the form the optimizer recognizes:

$$P_{required} - P_{available} < 0 \tag{10-6}$$

and in normalized form:

$$P_{required} / P_{avalable} - 1 < 0 \tag{10-7}$$

11. ANALYSIS DEMONSTRATION

This chapter verifies the relative accuracy of the analysis module by comparing the calculated results with known data from an existing general aviation aircraft. Particular attention has been given to the areas of stability and control and mission performance.

11.1 Cessna 182 Baseline

The Cessna 182 was selected for comparison, as there was substantial data available from a number of sources (RO95, SI71). Every parameter required for analysis was matched or approximated to closely replicate the actual aircraft. The following tables give the dimensions of the major design variables and parameters used:

TABLE 11.1.1: Cessna 182 Design Variables

DESIGN VARIABLE	VALUE
B (ft)	35.83
CROOT (ft)	5.73
TR	0.695
DIH (deg)	1.73
Xlewing (ft)	5.626
BT (ft)	11.54
Crtail (ft)	4.066
TRT	0.66
Xletail (ft)	21.6
BV (ft)	5.75
Crvt (ft)	3.9748
TRvt	0.62499

DESIGN VARIABLE	VALUE
Xlevt (ft)	22.13
BA (ft)	8.9
CA (ft)	0.75
YI (ft)	8.34
BE (ft)	11.53
CHE (ft)	1.44
BR (ft)	5.75
CR (ft)	1.2087
LB (ft)	25.0
Pmax (hp)	230
Wguess (lb)	2650.0
XM (ft)	7.35

TABLE 11.1.2: Cessna 182 Preassigned Parameters

PREASSIGNED PARAMETER	VALUE
Lo (ft)	25
Ho (ft)	4.85
H1o (ft)	4.8
H2o (ft)	1.8
HBCYo (ft)	2.9
HFCYo (ft)	3.5
Hgearo (ft)	4.5
HNOSEo (ft)	2.7
LBCYo (ft)	12.83
LFCYo (ft)	3.12
LMHo (ft)	14.5
R1o (ft)	0.73
WFUSo (ft)	3.6
WBCYo (ft)	3.1
WFCYo (ft)	3.6
WNOSEo (ft)	2.8
CGaco (ft)	5
CGaviono (ft)	6
CGctrlo (ft)	10
CGeleco (ft)	6
CGengo (ft)	2.94
CGfuelo (ft)	7.84
CGfurno (ft)	9
CGhydro (ft)	10
Slender	5
Option	1
SA (deg)	0
IWING (deg)	1.5
toverc	0.15
SAH (deg)	0
ITAIL (deg)	-3
SAvt (deg)	10
EngineDref (in)	7
EngineLref (in)	7
CD0	0.0279
CF	0.003

PREASSIGNED PARAMETER	VALUE
cLmaxClean	1.585
EnginDragFactor	1.1
EFF	0.85
ETAV	0.85
groundeffect	0.7
LDmax	12
Oswald	0.7
AnPower	1
enginePowerref	1
Nen	1
Wen	272
Dprop (in)	20
Npassengers	1
Ncrew	1
Wppass (lb)	200
Wpcrew (lb)	200
Wpayload (lb)	300
Growth	1
Wuav (lb)	50
RoskamA	0.8222
RoskamB	0.805
RaymerA	2.36
RaymerC	-0.18
HtHv	0
Nl (g's)	4.5
Nstrut	2
Fueldens (lb/gal)	6
Nt	2
Nz (g's)	3.9
Pdelta (psi)	0
Vpr (ft^3)	0
Wl (lb)	2800
Eps	0.05
ZA (ft)	1.67
ZT (ft)	0
ZV (ft)	2.82

TABLE 11.2: Cessna 182 Preassigned Parameters (continued)

CDA2DW (1/deg)	0	ZW (ft)	-1.835
CDFlapRatio	0.1	ALPHA (deg)	0
CDLGratio	0.1	GAMMA (deg)	0
cFriction	0.006	COSXZ	0.9981
CLA2DT (1/deg)	0.1	SINXZ	0.06105
CLA2DW (1/deg)	0.103	UpperMach	0.25
CLAFUS (1/deg)	0.0525	WindMax (ft/s)	100
CLmax	2.08	GRAVIT (ft/s^2)	32.2

The above values are a combination of published data and approximations based on the dimensions of Figure 2.1.

11.2 Cessna 182 Mission Profile

In order to obtain accurate results for mission sizing a typical mission profile was developed. General aviation aircraft, such as the Cessna 182, are used primarily for recreational flying or flying from point A to point B. As such, the mission profile developed for this analysis consists of takeoff from sea level, climb to a cruising altitude of 3,000 feet, cruise out 900 miles, descend and land at sea level. The results obtained using this simplified mission profile closely match published performance data.

11.3 Static Stability and Control

Much of the static and dynamic stability analysis is focused on the determination of the non-dimensional stability derivatives. The analysis performed on this aircraft shows that the analysis module closely approximates all of the desired stability derivatives. The

resulting derivatives for both the longitudinal and lateral-directional modes are presented against published data in the following table. Again, the analysis methods are approximations and are not perfect, however the results provide valuable insight at this stage of design.

TABLE 11.3.1: Non-dimensional Stability Derivative Comparison

Derivative	Cessna 182 Data	Results	% Difference
C_L	0.307	0.274678	-11.77%
C_D	0.032	0.031226	-2.48%
C_{L_α}	4.41	4.604820	4.23%
C_{D_α}	0.121	0.111509	-8.51%
C_{m_α}	-0.613	-0.673745	9.02%
$C_{L\dot{\alpha}}$	1.7	1.833249	7.27%
C_{Lq}	3.9	3.330721	-17.09%
C_{mq}	-12.4	-13.235016	6.31%
$C_{L\delta_E}$	0.43	0.409288	-5.06%
$C_{m\delta_E}$	-1.122	-1.272926	11.86%
$C_{y\beta}$	-0.393	-0.260722	-50.74%
$C_{l\beta}$	-0.0923	-0.084589	-9.12%
$C_{n\beta}$	0.0587	0.052965	-10.83%
C_{yp}	-0.075	-0.035188	-113.14%
C_{lp}	-0.484	-0.467007	-3.64%
C_{np}	-0.0278	-0.040513	31.38%
C_{y_r}	0.214	0.195577	-9.42%
C_{l_r}	0.0798	0.086677	7.93%
C_{n_r}	-0.0937	-0.094119	0.45%
$C_{l\delta_A}$	0.229	0.195850	-16.93%
$C_{n\delta_A}$	-0.0216	-0.016494	-30.96%

TABLE 11.3.1: Non-dimensional Stability Derivative Comparison (continued)

$C_{y_{\delta_R}}$	0.187	0.187369	0.20%
$C_{l_{\delta_R}}$	0.0147	0.014747	-0.46%
$C_{n_{\delta_R}}$	-0.0645	-0.067091	7.01%

Though actual data concerning control surface deflection requirements was not found, the results of the control analysis were within typical values for this type of aircraft. The following table summarizes these results:

TABLE 11.3.2: Control effectiveness results

CONDITION	DEFLECTION REQUIRED
Static Elevator Control	δ_E = -21.05 deg
Elevator Control for Takeoff Rotation	δ_E = -33.46 deg
Aileron Control for Static Roll Rate of 5 deg/s	δ_A = 9.85 deg
Control in a Crosswind (β = 10 deg)	δ_R = -6.23 deg δ_A = 5.68 deg

11.4 Dynamic Modes of Motion

The dynamic modes of motion are characterized by their response roots, natural frequencies, and damping ratios. Comparing these results with the known roots of the actual aircraft indicates that the analysis module closely approximates the actual dynamic stability of the given design. For this comparison, the analysis module has been allowed to calculate its own values for the moments of inertia. These calculations are highly

dependent on the selection of an appropriate radius of gyration (RA92 page 444), which were modified for this particular case.

TABLE 11.4.1: Dynamic Modes

	Cessna 182 Data	Results	% Difference
Short Period			
Roots	-4.13 ± 4.39	-4.22445 ± 4.92147	
ω_n (rad/s)	6.027	5.7209	-5.35%
ζ	0.685	0.53307	-28.50%
Phugoid			
Roots	-0.02092 ± 0.1797	-0.01227 ± 0.18053	
ω_n (rad/s)	0.181	0.2069	12.52%
ζ	0.116	0.07852	-47.73%
Dutch Roll			
Roots	-0.6858 ± 3.306	-0.13312 ± 3.92876	
ω_n (rad/s)	3.377	3,9101	99.99%
ζ	0.203	0.03386	-499.53%
Roll Mode			
Root	-12.43	-2.05062	-506.16%
τ (1/s)	0.080	0.48766	83.60%
Spiral Mode			
Root	-0.01095	-0.03576	69.38%

11.5 Performance Specifications

The results in this section are a product of a series of power and weight optimization test cases. For this analysis the propeller efficiency was assumed to be 0.85. The brake-specific-fuel consumption was set at 0.4/hr for the loiter legs and 0.5/hr for the cruise legs as suggested by RA92, page 19. Using the defined mission profile the performance results are as follows:

TABLE 11.5.1: Performance Specifications

	Cessna 182 Data	Results
Gross Weight (lb)	2800	2774.6
Empty Weight (lb)	1580	1580.01
Pmax (hp)	230	224.18
Fuel Used (lb)	360	373.79
Payload Weight (lb)	820	820
Takeoff Speed (ft/s)	96.8	96.60
Max Speed (ft/s)	245	225.88

12. TEST CASES AND RESULTS

Results of integrated design optimization studies of a general aviation airplane are presented in this chapter. These studies are conducted to gain initial insights and better understanding of how static and dynamic stability concerns, combined with mission performance and sizing issues, affect the resulting aircraft design. The chapter opens with a definition of the constraints, objectives and design variables used in the test cases discussed. The figures presented in each section present a rough, graphical representation of the resulting designs. The axis units are in feet.

12.1 Side Constraints and Behavior Constraints

Lower and upper bounds on design variables, and limits on performance, static stability and control as well as dynamic stability are listed in the following tables. The values for the constraints on the dynamic flight mechanics poles have been taken from requirements given by MIL-F-8785C.

TABLE 12.1.1: Side Constraints on design variables

DV	Lower	Upper
B (ft) wing span	10	60
CROOT (ft) wing root chord	1	10
TR wing taper ratio	0.5	1
DIH (deg)	0	10

DV	Lower	Upper
Xlevt (ft) x location of LE of vertical tail	10	30
BA (ft) aileron span	1	20
CA (ft) aileron chord	0.5	10
YI (ft) y-location inboard end of aileron	1	20

TABLE 12.1.1: Side Constraints on design variables (continued)

Xlewing (ft) x location of wing LE	3	20	BE span of elevator(ft)	1	25
BT (ft) span of horizontal tail	1	50	CHE chord of elevator(ft)	1	5
Crtail (ft) root chord of horizontal tail	1	10	BR span of rudder (ft)	1	20
TRT Horizontal tail taper ratio	0.5	1	CR chord of rudder (ft)	1	5
Xletail (ft) x location of horizontal tail LE	10	30	LB (ft) length of body	20	35
BV (ft) span of vertical tail	1	20	Pmax (hp) max engine power at sea level	10	500
Crvt (ft) Root chord of vetical tail	1	10	Wguess (lb) take-off gross weight	100	5000
TRvt taper ratio of vertical tail	0.5	1	XM x location of CG (ft)	4	20

TABLE 12.1.2: Stability and Control Behavior Constraints*

Constraint	Limit	Constraint	Limit
SMmax (static margin)	3	DRmax (deg) (rudder travel)	25
SMmin (static margin)	0.02	DRmin (deg) (rudder travel)	25
CYBo (cYbeta)	-1	PhDmin (phugoid damping)	0.04
CLBo (clbeta)	-0.2	SPDmin (short period damping)	0.3
CNBo (cnbeta)	0.3	SPDmax (short period damping)	2
DEmax (deg) (elevator travel)	25	SPnatmax (rad/s) (short period natural frequency)	3.6
DEmin (deg) (elevator travel)	25	SPnatmin(rad/s) (short period natural Frequency)	0.28
DAmax (deg) (aileron travel)	10	DRdmin (dutch roll damping)	0.08
DAmin (deg) (aileron travel)	10	Trollmax(sec) (roll root time constant)	1.4

* upper and lower bounds on control surface travel and flight mechanics poles will be changed in order to study the effect on the resulting designs.

12.2 Case 1: Sizing for Mission and Performance

The purpose of this simple case was to test the functionality of the combined optimizer-analysis program. The following mission profile was used for this analysis:

TABLE 12.2.1: Sizing Mission Definition

(The present synthesis capability allows any number of mission segments in the mission definition)

Altitude/Dh (ft)	1000	3000	3000	3000	3000	3000	0
Distance/Time (miles) or (min)	0	0	200	10	200	10	0
BSFC (lbf/hr/hp)	0.4	0.4	0.4	0.5	0.4	0.5	0.4
Npropeller (prop efficiency)	0.8	0.8	0.8	0.7	0.8	0.7	0.8
Airspeed (ft/s)	100	0	210	130	220.1	130	75
Throttle	1	0.85	0.8	0.7	0.8	0.7	0.4
Runway length (ft)	1450	0	0	0	0	0	5000
ROC Req. (fps) rate of climb	10	10	10	10	10	10	200
Obstacle ht (ft) for t/o and landing	50	0	0	0	0	0	50
	Takeoff	Climb	Cruise	Loiter	Cruise	Loiter	Landing
Mission segment	1	2	3	4	5	6	7

Two different initial designs were tried, and the objective function to be minimized was the aircraft gross take-off weight. The design variables used were the wing span, the maximum power, and the take-off gross weight. Constraints for each mission leg were imposed, guaranteeing enough power at each segment, service ceiling, and take-off and landing distances. An equality constraint is used to force a match between the guessed weight (at the beginning of a new mission analysis) and the resulting weight (after fuel

weight is calculated for the mission). Since the new capability allows for any number of mission segments in a mission, and since each mission segment adds performance constraints to the set of constraints, the size of the constraint vector varies depending on the number of mission legs used. This test case is defined by:

Design Variables Used: B, P_{max}, W_{guess}

Constraints Used: C(52-73); Ceq(5)

Objective Function: Minimize $Wcalc$

The unused design variables and the pre-assigned parameters were set equal to the Cessna 182 values. This practice is maintained for subsequent cases, as well. The initial designs were started from extremes on either side of the resulting design. Under the same restrictions, the optimum designs resulting from both initial designs were nearly identical, and Table 12.2.2 summarizes the results. Note how the rate of climb during the second cruise leg drove the amount of power required to meet this constraint.

TABLE 12.2.2: Case 1 Results (Design Variables)

DV	Initial #1	Result #1	Initial #2	Result #2
B (ft)	10	35.28	60	35.3
Pmax (hp)	10	64.12	200	64.10
Wguess (lb)	500	1458.73	4000	1458.73

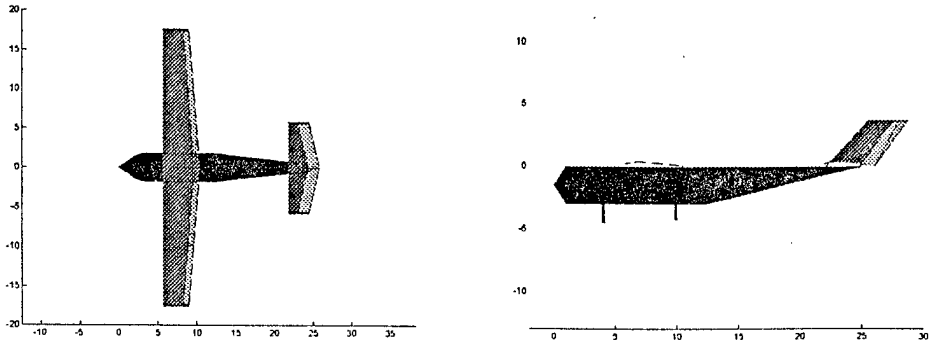


Figure12.1. Case 1 Design Plot

TABLE 12.2.3: Case 1 Results vs. Requirements (Behavior Constraints)

Constraint	Required	Result
Takeoff Distance (ft)	1450	1403.57
Takeoff Speed (ft/s)	100	70.75
Power Req. @ Takeoff (hp) *	4.3980	51.2925
ROC @ Takeoff (ft/s)	10	11.76
ROC @ Cruise (leg1 / leg2) (ft/s)	10 / **10**	10.78 / **10.00**
Power @ Cruise (leg1 / leg2) (hp) *	22.35 / 22.19	37.54 / 35.35
Power @ Loiter (leg1 / leg2) (hp) *	18.97 / 18.27	28.74 / 22.19
Landing Distance (ft)	5000	3038.85
Landing Speed (ft/s)	75	41.69

* Indicates a calculated requirement

12.3 Case 2: Designing with Added Static Stability and Control Constraints

12.3.1 Initial Test Case

This case focuses on obtaining a design that meets the constraints imposed on the aircraft's static stability and controllability. This is in addition to the mission performance and sizing restrictions used in Case 1. Static stability and control are primarily functions of the lifting surfaces (wing, horizontal tail, and vertical tail), and the

control surfaces (elevator, ailerons, and rudder), and center of gravity location. As such, the surface defining design variables have been included for this case. In Test Case 2 here it is assumed that the span of the entire elevator is equal to the span of the horizontal tail. The same assumption was made concerning the span of the rudder and the span of the vertical tail.

Initial attempts to synthesize an airplane for this case were hampered by the optimizer's inability to resolve the requirement on the amount of elevator required for takeoff rotation ($\delta_{E_{rotate}}$) with the requirement for a desirable static margin. For this reason, the center of gravity position was also added as a design variable. This case is defined by:

Design Variables Used: *B*, *BT*, *BV*, *BA*, *CA*, *CHE*, *CR*, P_{max}, W_{guess}, *XM*

Constraints Used: *C*(1, 3, 4, 7-9, 16-44, 52-73); *Ceq*(5)

Objective Function: Minimize *Wcalc*, *SELEV*, *SR*, and *SAileron*

The side constraints and behavior constraints for this case have been unchanged. The mission profile used is the same mission defined in Chapter 11 for the purposes of matching the Cessna 182 performance specifications. The initial design sent to the optimizer and the resulting design are presented in the following tables.

TABLE 12.3.1: Case 2 Resulting Design (Design Variables)

DV	Initial	Result
B (ft)	30	34.1341
BT (ft)	10	11.9672
BV (ft)	5	5.23
BA (ft)	2	4.0731
CA (ft)	1	0.9788

DV	Initial	Result
CHE (ft)	1	1.1020
CR (ft)	1	1.0980
Pmax (hp)	50	224.18
Wguess (lb)	1000	2774.6
XM (ft)	6	7.7614

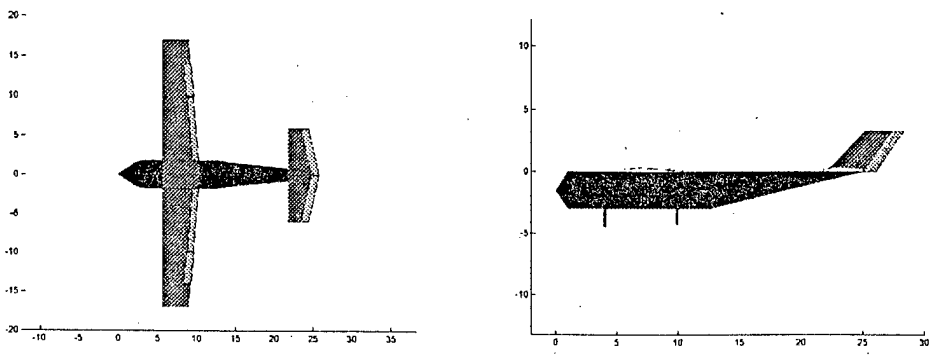


Figure 12.2. Case 2 Design Plot

TABLE 12.3.2: Case 2 Results vs. Requirements (Behavior Constraints)

Constraint	Required	Result
Static Margin	0.02 < SM < 3	0.0412
C_{y_β}	-1 < C_{y_β} < 0	-0.2250
C_{l_β}	-0.2 < C_{l_β} < 0	-0.0803
C_{n_β}	0 < C_{n_β} < 0.3	0.0267
$\delta_{E_{static}}$ (deg)	-25< δ_E < 25	-4.1880
$\delta_{E_{rotate}}$ (deg)	-25< δ_E < 25	-21.792
$\delta_{A_{static}}$ (deg)	-10< δ_A < 10	6.4145
$\delta_{A_{crosswind}}$ (deg)	-10< δ_A < 10	9.8503
$\delta_{R_{crosswind}}$ (deg)	-25< δ_R < 25	-24.1995

This result was completed successfully and met all the requirements imposed. A comparison of this result to the dimensions of the Cessna 182 shows that airplane synthesized here is nearly the same as the Cessna 182.

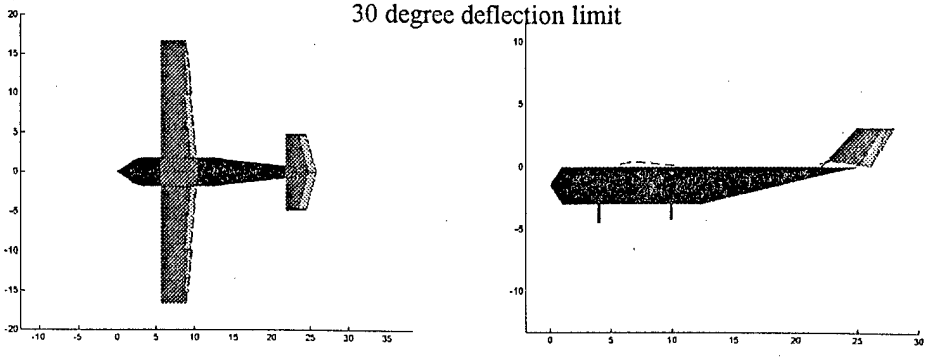
12.3.2 Variations of Case 2

This section summarizes a series of variations to the requirements imposed on Case 2. The question was asked, what happens to the design when the available control surface deflection is increased or reduced? This question was studied by imposing control surface deflection limits of 30, 20, and 10 degrees on each control surface simultaneously in a sequence of synthesis runs. As was expected, the larger the amount of available deflection, the less control surface area was needed. The following results represent the observations made for this series of runs. Note how the elevator travel for take-off rotation and the rudder travel for cross-wind landing drive the designs. The same initial design as in Case 2 above was used.

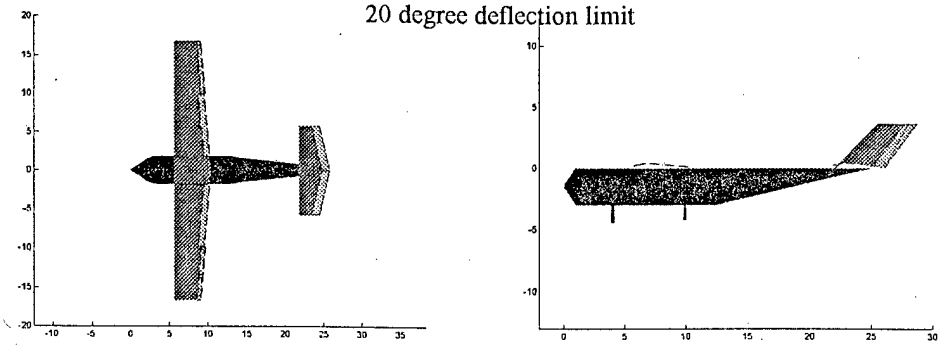
TABLE 12.3.3: Effects of control-deflection travel-limits on final design (Design Variables)

DV	Initial	30 deg travel limit	20 deg travel limit	10 deg travel limit
B (ft)	30	33.4010	33.4010	33.6376
BT (ft)	10	9.6312	11.6559	15.3558
BV (ft)	5	4.7598	5.6854	8.0860
BA (ft)	2	5.3133	6.7373	8.1689
CA (ft)	0.5	0.5	0.5	0.5
CHE (ft)	1	1.0	1.0	1.0
CR (ft)	1	1.0	1.0	1.0
Pmax (hp)	50	238.6628	240.7075	232.5183
Wguess (lb)	1000	2715.0210	2715.0210	2715.0210
XM (ft) CG location	6	7.6984	8.0013	8.5394

30 degree deflection limit



20 degree deflection limit



10 degree deflection limit

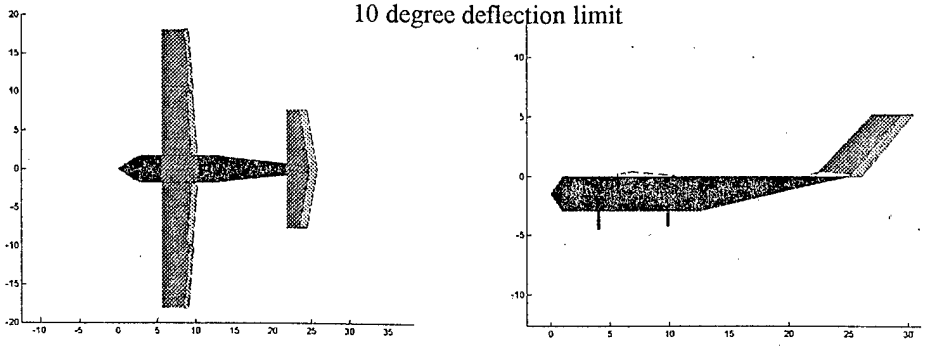


Table 12.3. Case 3 Design Plot Comparison

TABLE 12.3.4: Effects of control-deflection travel-limits on behavior constraint values

Constraint	Required	30 deg	20 deg	10 deg
Static Margin	0.02 < SM < 3	0.0200	0.0200	0.0200
C_{y_β}	$-1 < C_{y_\beta} < 0$	-0.2034	-0.2842	-0.5155
C_{l_β}	$-0.2 < C_{l_\beta} < 0$	-0.0850	-0.0912	-0.1083
C_{n_β}	$0 < C_{n_\beta} < 0.3$	0.0235	0.0556	0.1458
$\delta_{E_{static}}$ (deg)	Limit varies	-2.4370	-1.9311	-1.4192
$\delta_{E_{rotate}}$ (deg)	Limit varies	-30.0000	-20.0000	-10.0000
$\delta_{A_{static}}$ (deg)	Limit varies	6.1828	7.2635	7.1485
$\delta_{A_{crosswind}}$ (deg)	Limit varies	11.9434	9.7163	9.2138
$\delta_{R_{crosswind}}$ (deg)	Limit varies	-30.0000	-20.0000	-10.0000

12.4 Case 3: Designing with Added Dynamic Stability Constraints

The cases from section 12.3.1 and 12.3.2 focused only on static stability and control requirements. This section extends these same cases into the realm of dynamic stability. The behavior constraints on the short period, phugoid, Dutch roll, and roll modes of motion now had to be added to the list of constraints. No design variables were added for this case. Thus, this case is defined by:

Design Variables Used: *B*, *BT*, *BV*, *BA*, *CA*, *CHE*, *CR*, P_{max}, W_{guess}, *XM*

Constraints Used: *C*(1, 3, 4, 7-9, 16-73); *Ceq*(5)

Objective Function: Minimize *Wcalc*, *SELEV*, *SR*, and *SAileron*

This side constraints and behavior constraints for this case have been unchanged except as indicated by each particular case. The mission profile used is the same mission

defined in Chapter 11 for the purposes of matching the Cessna 182 performance specifications. Presented in the following tables are the new designs and results obtaining using the same three cases for control surface travel limits as in section 12.3.2. The static stability results are used to allow comparison between the results and those of section 12.3.2. In Table 12.4.2, the values in parentheses indicate the value found for the original designs of Case 2. Due to the similarities between this case and Case 2, a design plot comparison is not given.

TABLE 12.4.1: Case 3 Resulting Designs (Design Variables)

DV	Initial	30 deg	20 deg	10 deg
B (ft)	30	38.2492	34.1341	34.1341
BT (ft)	10	10.9719	12.1839	15.7278
BV (ft)	5	4.1859	4.1943	4.3047
BA (ft)	2	1.5388	2.2737	4.5499
CA (ft)	0.5	0.5	0.5	0.5
CHE (ft)	1	1.0	1.0	1.0
CR (ft)	1	1.0	1.0	1.0
Pmax (hp)	50	209.0204	280.6459	242.6341
Wguess (lb)	1000	2715.0210	2715.0210	2715.0210
XM (ft)	6	7.6784	7.9501	8.4042

TABLE 12.4.2: Case 3 Results vs. Requirements (Behavior Constraints)

Constraint	Required	30 deg	20 deg	10 deg
$\zeta_{phugoid}$	> 0.04	0.08186 (0.0794)	0.0770 (0.0792)	0.0768 (0.2069)
$\zeta_{ShortPeriod}$	$0.3 < \zeta_{SP} < 2.0$	1.68877 (1.6259)	2.0000 (2.0263)	2.0000 (2.6792)
$\omega_{n_{ShortPeriod}}$ (rad/s)	$0.28 < \omega_n < 3.6$	1.7504 (1.6767)	1.7085 (1.6737)	2.2397 (1.6736)
$\zeta_{DutchRoll}$	>0.08	0.1154 (0.2562)	0.47816 (0.2228)	0.0965 (0.2299)
τ_{Roll} (sec)	< 1.4 sec	0.5073 (0.4163)	0.4689 (0.4439)	0.4785 (0.4922)
Static Margin	0.02 < SM < 3	0.0200	0.0214	0.036953
C_{y_β}	$-1 < C_{y_\beta} < 0$	-0.1322	-0.1539	-0.1625

TABLE 12.4.2: Case 3 Results vs. Requirements (Behavior Constraints) (continued)

C_{l_β}	$-0.2 < C_{l_\beta} < 0$	-0.0704	-0.0815	-0.0822
C_{n_β}	$0 < C_{n_\beta} < 0.3$	0.0000	0.000361	0.0000001
$\delta_{E_{static}}$ (deg)	Limit varies	-2.3636	-1.9801	-2.5585
$\delta_{E_{rotate}}$ (deg)	Limit varies	-30.0000	-19.9999	-10.0000
$\delta_{A_{static}}$ (deg)	Limit varies	0.0495	0.1022	0.2066
$\delta_{A_{crosswind}}$ (deg)	Limit varies	30.0000	20.0000	10.0000
$\delta_{R_{crosswind}}$ (deg)	Limit varies	-1.6658	-1.8666	-1.9675

Note how, again, the limits on control surface travel (elevator and rudder) drive the design. But now additional constraints, due to requirements on frequency and damping of airplane dynamic modes, become active. Most noticeable is the importance of short-period damping coefficient. Also note the variation of Static Margin with control surface travel limit. This is different from the cases presented in section 12.3.2, where, in an effort to minimize control surface travel the optimization always drove the static margin to a minimum.

12.5 Case 4: Added Fuselage Sizing / Surface Positioning

The final case presented here is an attempt to add aircraft fuselage length and positions of the lifting surfaces to the set of design variables. Also added are positions of landing gear components, the length of the main landing gear, the center of gravity location, and constraints on ground handling stability. The dimensions of the wing, horizontal tail and vertical tail are set constant to the Cessna 182 values. The width and height dimensions

of the fuselage were scaled according to the length of the fuselage. This case is defined by:

Design Variables Used: *Xlewng, Xletail, Xlevt, LB, CGmgr, Lm, CGngr,* P_{max}, W_{guess}, *XM*

Constraints Used: *C*(10-17, 24-44, 52-73); *Ceq*(1-5)

Objective Function: Minimize a combination Wcalc *(min weight)* and *LB (min size)*

The side constraints and behavior constraints for this case have been unchanged from Case 3 except as indicated. The mission profile used is the same mission defined in Chapter 11 for the purposes of matching the Cessna 182 performance specifications. The following tables present the resulting design and the subsequent results. Note that Table 12.5.2 contains control surface sizing results. A second case, similar to Case 2, was run using the results of the current case to size the control surfaces based on static stability criteria.

TABLE 12.5.1: Case 4 Resulting Design (Design Variables)

DV	Initial	Result
Xlewng (ft)	5.5	6.2560
Xletail (ft)	21	13.9794
Xlevt (ft)	21	14.0252
LB (ft)	25.75	18.0000
CGmgr (ft)	9.8	8.4914
CGngr (ft)	3.92	1.6646
Pmax (hp)	50	239.3645
Wguess (lb)	1000	2715.0210
XM (ft)	6	7.8087
BA (ft)	2	2.2402
CA (ft)	0.5	0.5
CHE (ft)	1	1
CR (ft)	1	1

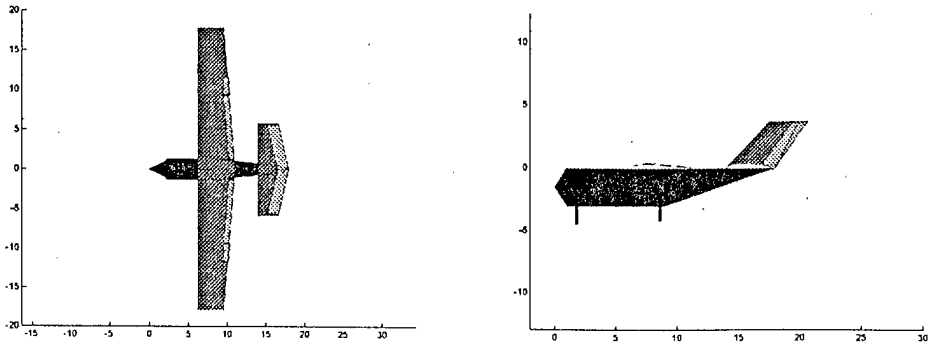


Figure 12.4. Case 4 Design Plot

TABLE 12.5.2: Case 4 Results vs. Requirements (Behavior Constraints)

Constraint	Required	Result
Tipangle (deg)	15	15.0000
Ground Clearance (deg)	15	15.0000
Nose Gear Loading (% Wguess)	10%	10.00%
Main Gear Loading (% Wguess)	90%	90.00%
Static Margin	$0.02 < SM < 3$	0.0251
C_{y_β}	$-1 < C_{y_\beta} < 0$	-0.2257
C_{l_β}	$-0.2 < C_{l_\beta} < 0$	-0.0826
C_{n_β}	$0 < C_{n_\beta} < 0.3$	0.0189
$\delta_{E_{static}}$ (deg)	Limit varies	-5.6794
$\delta_{E_{rotate}}$ (deg)	Limit varies	-19.2481
$\delta_{A_{static}}$ (deg)	Limit varies	0.0879
$\delta_{A_{crosswind}}$ (deg)	Limit varies	20.0000
$\delta_{R_{crosswind}}$ (deg)	Limit varies	-4.6261

13. CONCLUSION

The development, validation, and initial testing of a new capability for design synthesis of general aviation airplanes has been described. The new capability is unique in its integration of mission analysis and performance considerations with static stability and control and dynamic stability requirements. The key to the new development is a collection of very fast, computationally efficient, techniques for aerodynamic, inertial (weight and balance), performance, mission, as well as static and dynamic stability and control analysis. This fast analysis makes it possible, using standard gradient-based nonlinear programming techniques, to synthesize optimum airplane configurations with tens of design variables and tens of constraints. Design variables which define the overall shape of the airplane, including size and location of tail surfaces and control surfaces, are allowed to change simultaneously in an effort to meet all constraints while minimizing weight or size or combinations of those. Other objective functions, made of combination of any behavior constraints included in the present capability, can be easily included. Through the course of utilizing this new capability, a number of areas were noticed which could benefit from some minor improvements:

a. Addition of a graphical user interface. The use of Lotus 1-2-3 worksheets allows this program to easily be extended to a Visual Basic-type GUI. This would allow the user to define and run a problem much more quickly.

b. A number of non-physical results may arise if the appropriate constraints and conditions are not properly imposed. Preferably the program should safeguard against such problems without adding undue problems for the user. One example can be noticed in Case 4. While the fuselage was minimized, no constraint exists to ensure that the internal fuselage volume remains large enough for crew, passengers, and payload.

c. Redundancies in the output files need to be addressed. The four current output files could easily be reduced to one or two.

d. The singular module that allows the user to plot a graphical picture of the final design has not yet been linked directly to the entire code. This would be preferable to having to run this module separately.

Future developments of the new capability call for:

a. Extension to more families of airplanes, including small UAVs;

b. Thorough studies of the interaction of performance and mission requirements with static and dynamic stability and control requirements with increasing numbers of constraints and design variables; and

c. Addition of active control systems, and development of a capability for combined optimal design of air vehicle and its active control system.

In its present form the new capability is a valuable addition to the computer design tools available for conceptual design of general aviation airplanes. It will be very useful for educational purposes, especially, in undergraduate and graduate airplane design courses.

REFERENCES

1. (AM60) Ashkenas, I.L., and McRuer, D.T., "Optimization of the Flight-Control, Airframe System", Journal of the Aerospace Sciences, March, 1960, pp. 197-218.
2. (BI49) Bird, John P., "Some Theoretical Low-Speed Span Loading Characteristics of Swept Wings in Roll and Sideslip," NACA TN-1839, March 1949.
3. (BL65) Blakelock, John H., Automatic Control of Aircraft and Missiles, John Wiley and Sons, Inc., NY, 1965.
4. (CM43) Campbell, John P. and Ward O. Mathews, "Experimental Determination of the Yawing Moment Due to Yawing Contributed by the Wing, Fuselage, and Vertical Tail of a Midwing Airplane Model," WR L-387, June 1943.
5. (CM52) Campbell, John P. and Marion O. McKinney, "Summary of Methods for Calculating Dynamic Lateral Stability and Response and for Estimating Lateral Stability Derivatives," NACA TR-1098, 1952.
6. (CO85) Covert, Eugene, E., Thrust and Drag: Its Prediction and Verification, AIAA, New York, 1985.
7. (DSC67) Dommasch, Daniel O., Sydney S. Sherby, and Thomas F. Connolly, Airplane Aerodynamics, Pitman Publishing, 1967.
8. (ET59) Etkin, Bernard, Dynamics of Flight, Stability and Control, John Wiley and Sons, Inc., NY, 1959.

9. (GA49) Goodman, Alex and Glenn H. Adair, "Estimation of the Damping in Roll of Wings through the Normal Flight Range of Lift Coefficient," NACA TN-1924, July 1949.

10. (GB48) Goodman, Alex and Jack D. Brewer, "Investigation of Low Speeds on the Effect of Aspect Ratio and Sweep on Static and Yawing Stability Derivatives of Untapered Wings," NACA TN-1669, August 1948.

11. (GF50) Goodman, Alex and Lewis R. Fisher, "Investigation at Low Speeds of the Effect of Aspect Ratio and Sweep on Rolling Stability Derivatives of Untapered Wings," NACA TR-968, 1950.

12. (GS44) Greenberg, Harry and Leonard Sternfield, "A Theoretical Investigation of Longitudinal Stability of Airplanes with Free Controls Including Effect of Friction in Control System," NACA TR-791, 1944.

13. (GW41) Gilruth, R. R. and M. D. White, "Analysis and Prediction of Longitudinal Stability of Airplanes," NACA TR-711, 1941.

14. (HE68) Hoak, D. E. and D. E. Ellison, "USAF Stability and Control Datcom," October 1960 (rev. August 1968).

15. (JA32) Jacobs, Eastman N., "Characteristics of Two Sharp-nosed Airfoils Having Reduced Spinning Tendencies," NACA TN-416, April 1932.

16. (JW35) Jacobs, Eastman N. and Kenneth E. Ward, "Interference of Wing and Fuselage from Tests of 209 Combinations in the N.A.C.A. Variable-Density Tunnel," NACA TR-540, 1935.

17. (KW57) Klawans, Bernard B. and Jack A. White, "A Method Utilizing Data on the Spiral, Roll-Subsidence, and Dutch Roll Modes for Determining Lateral Stability Derivatives for Flight Measurements," NACA TN-4066, August 1957.

18. (LR80) Lan, Chuan-Tau Edward and Jan Roskam, Airplane Aerodynamics and Performance, RAEC, Lawrence, 1980.

19. (MC95) Mayo, William E. and Martin Cwiakala, Programming with Fortran 77, McGraw-Hill, N.Y., 1995.

20. (MI80) MIL-F-8785C. "Military Specification: Flying Qualities of Piloted Airplanes," 5 Novemeber 1980.

21. (MK90) Morris, S., Kroo, I., "Aircraft Design Optimization with Dynamic Performance Constraints," Journal of Aircraft, Vol. 27, No. 12, Dec. 1990, pp. 1060-1067.

22. (MO92) Morris, S., "Integrated Aerodynamic and Control System Design of Tailless Aircraft", Proceedings of the AIAA Guidance, Navigation, and Control Conference, AIAA, Washington, DC, August 1992.

23. (PH49) Perkins, Courtland D. and Robert E. Hage, Airplane Performance Stability and Control, John Wiley and Sons, Inc., NY, 1949.

24. (PJ38) Pearson, Henry A. and Robert T. Jones, "Theoretical Stability and Control Characteristics of Wings with Various Amounts of Taper and Twist," NACA TR-635, 1938.

25. (QU56) Queijo, M. J., "Theoretical Span Load Distribution and Rolling Moments for Sideslipping Wings of Arbitrary Plan Form in Incompressible Flow," NACA TR-1269, 1956.

26. (RA92) Raymer, Daniel P., Aircraft Design: A Conceptual Approach, 2nd ed., AIAA, Washington, DC, 1992.

27. (RO185) Roskam, Jan, Airplane Design: Part II, Preliminary Configuration Design and Integration of the Propulsion System, RAEC, Lawrence, 1985.

28. (RO285)Roskam, Jan, Airplane Design: Part V, Component Weight Estimation, RAEC, Lawrence, 1985.

29. (RO95) Roskam, Jan, Airplane Flight Dynamics and Automatic Flight Controls, Part I, DARcorp, Lawrence, 1995.

30. (SC98) Schmidt, Louis V., Introduction to Aircraft Flight Dynamics, AIAA, Reston, 1998.

31. (SM84) Smetana, Frederick O., Computer Assisted Analysis of Aircraft Performance Stability and Control, McGraw-Hill, N.Y., 1984.

32. (TO48) Toll, Thomas A., and M. J. Queijo, "Approximate Relations and Charts for Low-Speed Stability Derivatives of Swept Wings," NACA TN-1581, 1948.

33. (TO82) Torenbeek, Egbert, Synthesis of Subsonic Airplane Design, Delft Univ. Press, Delft, Holland, 1982.

34. (UM99) User's Manual, MATLAB Student Version 5.3, MathWorks, Inc., 1999.

35. (VA99) Vanderplaats, Garret N., Numerical Optimization Techniques for Engineering Design, 3rd ed., VRAND, Colorado Springs, 1999.

36. (VO45) Von Mises, Richard, Theory of Flight, McGraw-Hill, NY, 1945.

37. (WI71) William, Silver B., "Optimization Studies in Aircraft Design," Dissertation, Stanford University, 1971.

38. (WO51) Wolhart, Walter D., "Influence of Wing and Fuselage on the Vertical Tail Contribution to the Low Speed Rolling Derivative of Midwing Airplane Models with 45° Sweptback Surfaces," NACA TN-2587, December 1951.

APPENDIX A

Analyze.m PROGRAM INFORMATION

A.1 Introduction

This appendix contains user information for Analyze.m, the analysis code developed for optimization in this thesis. The program subroutines are briefly described in the order in which they are executed. This program utilizes the global nature of variables and parameters within MATLAB (UM99).

A.2 Program Subroutines

The main program, Analyze.m, calls the following subroutines, presented in order of appearance, to perform the indicated functions.

NAME	FUNCTION
Geometry.m	Read and assign input files. Perform basic geometric calculations.
Fuselage.m	Scale and calculate fuselage dimensions.
Parasite.m	Analytically determine the parasite drag coefficient.
Mission.m	Read in mission definition. Organize and conduct mission analysis.
Takeoff.m	Perform takeoff analysis.
Atmos.m	Determines atmospheric conditions at any given altitude.
Maxspeed.m	Determines maximum speed at any given altitude
Climb.m	Orchestrate climb analysis.
BestClimb.m	Perform optimum climb analysis.
Cruise.m	Perform cruise analysis.
Loiter.m	Perform loiter analysis.
Landing.m	Perform landing analysis.

Weights.m	Perform weights analysis and determine moments of inertia.
LandingGear.m	Determine ground handling stability parameters.
VdocLongStab.m	Perform longitudinal static stability analysis.
VdocLongDynm.m	Perform longitudinal dynamic stability analysis.
VdocLatDir.m	Perform lateral directional static stability analysis.
VdocLatDynm.m	Perform lateral directional dynamic stability analysis.
Controls.m	Determine control effectiveness requirements.
Allconstraints.m	Generate list of all constraints.

A.3 Input Files

Due to the significant flexibility afforded this program, the number of input files is quite large. The first three allow the user to define the variables and parameters. The remaining files allow the user to cater the program to their particular design needs. Each file must be saved as a LOTUS 1-2-3 worksheet.

NAME	CONTENTS
variables.wk1	The set of user defined design variables.
usedvars.wk1	Determines which design variables are used for optimization (1 = used; 0 = not used)
lower.wk1	Lower bounds on design variable values.
upper.wk1	Upper bounds on design variable values.
parameters.wk1	The set of user defined permanent parameters.
mission.wk1	User defined mission profile
UserConstraints.wk1	User defined limits on the constraints.
ConstraintsUsed.wk1	Determines which inequality constraints are used (1 = used; 0 = not used)
EqconsUsed.wk1	Determines which equality constraints are used (1 = used; 0 = not used)

APPENDIX B

USER'S MANUAL

B.1 Introduction

The purpose of this user's manual is to provide enough guidance to allow a new user to learn how to interact with the complete program and perform new design cases. This information includes the format of the input files, how to utilize these files to create new design cases, and how to run the program and interpret the results. Each section presents a series of steps to be taken to originate a new design from scratch. These are continuous through each section.

B.2 Managing the input files

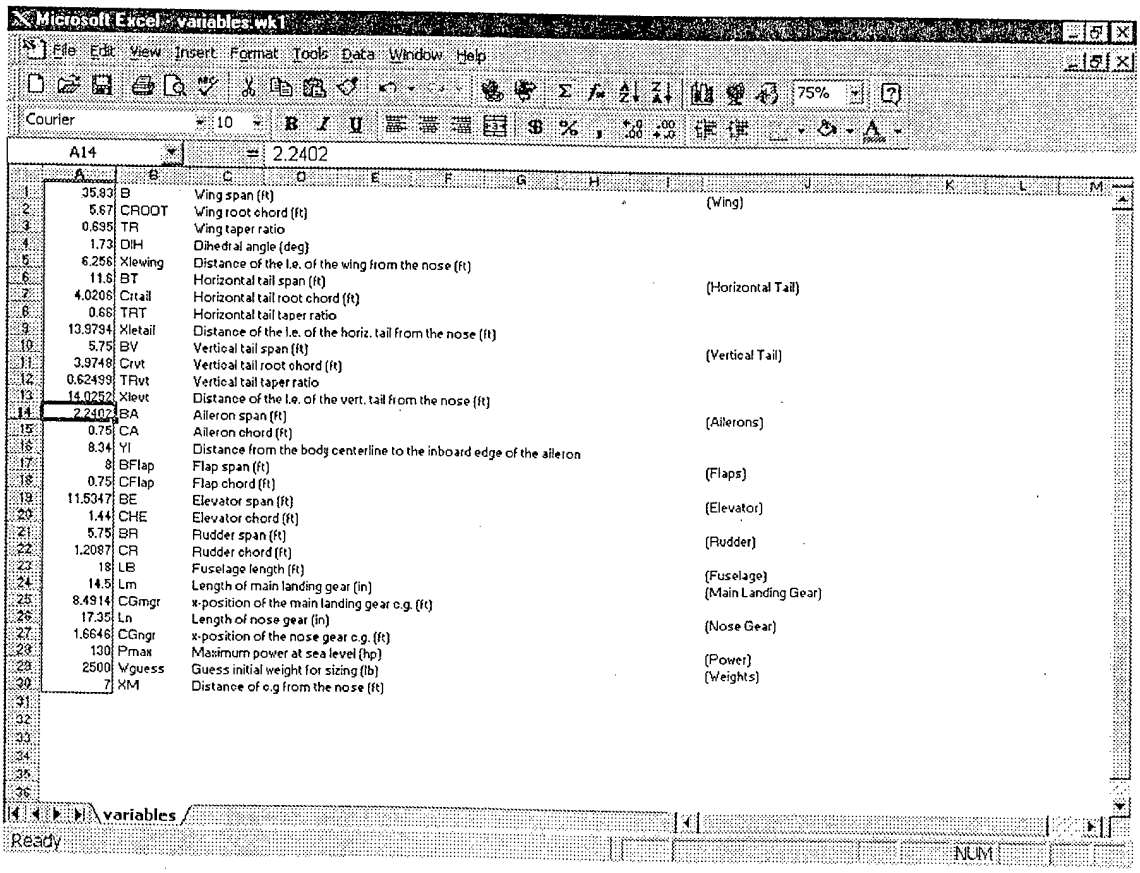
In Appendix A, the names and functions of the individual input files were presented. These files are the primary manner in which the user defines the optimization design problem of interest.

Step #1: Open all input files

The input files are in LOTUS 1-2-3 worksheet format and can be opened and saved using Micosoft *Excel*. Open each of the input files so that you can toggle between the worksheets using the **Window** command on the *Excel* toolbar. There should be nine files. Check section A.3 of Appendix A to verify that all the files were opened properly.

***Step #2**: Define your initial design using* **variables.wk1** *and* **parameters.wk1**

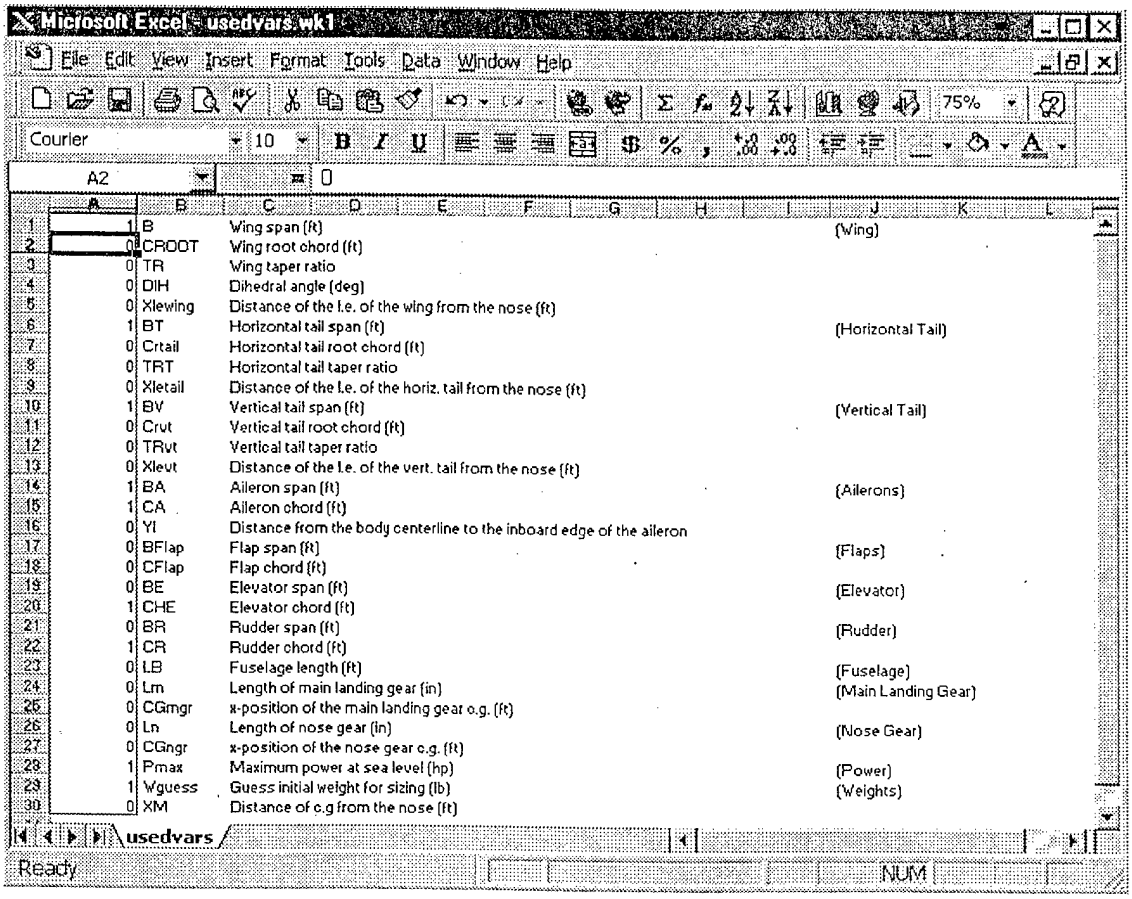
These files define the initial design. The preassigned parameters that you indicate will be fixed throughout your design analysis. Unused design variables will also be fixed and should be set to reasonable or known values. The design variables used for optimization can be selected at the user's discretion. The following screen-shot shows **variables.wk1** as it should appear. The user defines the values in column A, highlighted in red. The same setup has been applied to **parameters.wk1**.



35.83	B	Wing span (ft)	(Wing)
5.67	CROOT	Wing root chord (ft)	
0.695	TR	Wing taper ratio	
1.73	DIH	Dihedral angle (deg)	
6.256	Xlewing	Distance of the l.e. of the wing from the nose (ft)	
11.6	BT	Horizontal tail span (ft)	(Horizontal Tail)
4.0206	Crtail	Horizontal tail root chord (ft)	
0.66	TRT	Horizontal tail taper ratio	
13.9794	Xletail	Distance of the l.e. of the horiz. tail from the nose (ft)	
5.75	BV	Vertical tail span (ft)	(Vertical Tail)
3.9748	Crvt	Vertical tail root chord (ft)	
0.62499	TRvt	Vertical tail taper ratio	
14.0252	Xlevt	Distance of the l.e. of the vert. tail from the nose (ft)	
2.2402	BA	Aileron span (ft)	(Ailerons)
0.75	CA	Aileron chord (ft)	
8.34	YI	Distance from the body centerline to the inboard edge of the aileron	
8	BFlap	Flap span (ft)	(Flaps)
0.75	CFlap	Flap chord (ft)	
11.5347	BE	Elevator span (ft)	(Elevator)
1.44	CHE	Elevator chord (ft)	
5.75	BR	Rudder span (ft)	(Rudder)
1.2087	CR	Rudder chord (ft)	
18	LB	Fuselage length (ft)	(Fuselage)
14.5	Lm	Length of main landing gear (in)	(Main Landing Gear)
8.4914	CGmgr	x-position of the main landing gear c.g. (ft)	
17.35	Ln	Length of nose gear (in)	(Nose Gear)
1.6646	CGngr	x-position of the nose gear c.g. (ft)	
130	Pmax	Maximum power at sea level (hp)	(Power)
2500	Wguess	Guess initial weight for sizing (lb)	(Weights)
7	XM	Distance of c.g from the nose (ft)	

Step #3*: *Using* usedvars.wk1, *define which design variables are to be used

This file simply defines a vector of ones and zeros to indicate which design variables are used for a particular case. The following screen shot illustrates a case where the variables, *B, BT, BV, BA, CA, CHE, CR, Pmax,* and *Wguess* are all designated as used design variables.



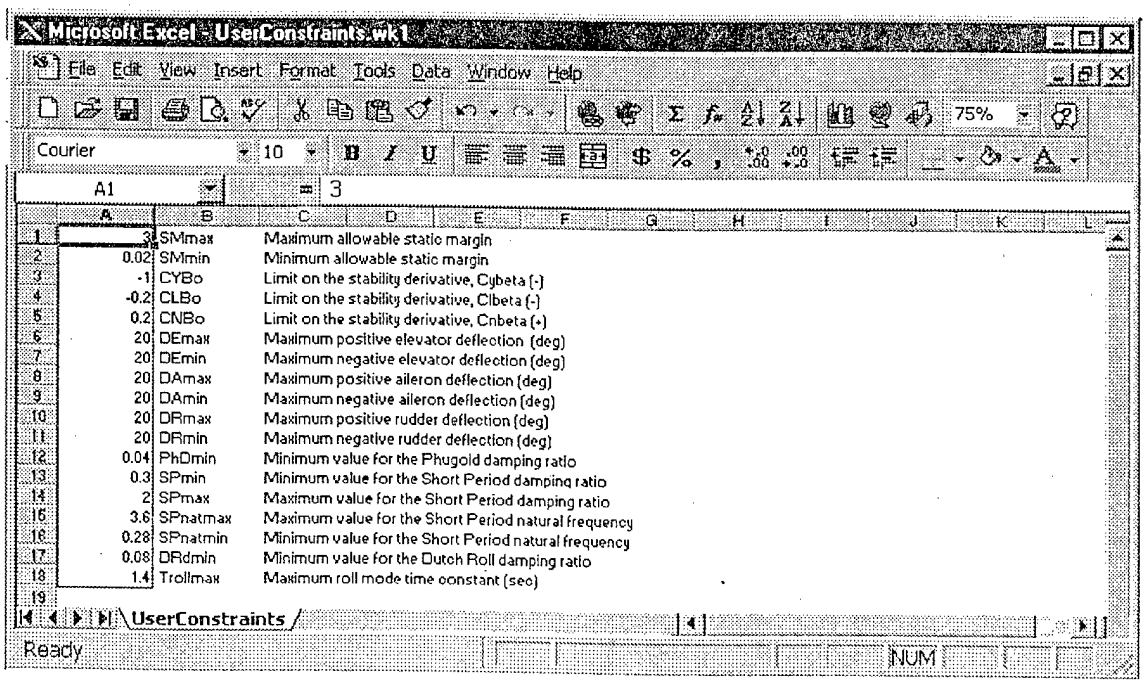
	A	B	C	J
1	1	B	Wing span (ft)	(Wing)
2	0	CROOT	Wing root chord (ft)	
3	0	TR	Wing taper ratio	
4	0	DIH	Dihedral angle (deg)	
5	0	Xlewing	Distance of the l.e. of the wing from the nose (ft)	
6	1	BT	Horizontal tail span (ft)	(Horizontal Tail)
7	0	Crtail	Horizontal tail root chord (ft)	
8	0	TRT	Horizontal tail taper ratio	
9	0	Xletail	Distance of the l.e. of the horiz. tail from the nose (ft)	
10	1	BV	Vertical tail span (ft)	(Vertical Tail)
11	0	Crut	Vertical tail root chord (ft)	
12	0	TRvt	Vertical tail taper ratio	
13	0	Xlevt	Distance of the l.e. of the vert. tail from the nose (ft)	
14	1	BA	Aileron span (ft)	(Ailerons)
15	1	CA	Aileron chord (ft)	
16	0	YI	Distance from the body centerline to the inboard edge of the aileron	
17	0	BFlap	Flap span (ft)	(Flaps)
18	0	CFlap	Flap chord (ft)	
19	0	BE	Elevator span (ft)	(Elevator)
20	1	CHE	Elevator chord (ft)	
21	0	BR	Rudder span (ft)	(Rudder)
22	1	CR	Rudder chord (ft)	
23	0	LB	Fuselage length (ft)	(Fuselage)
24	0	Lm	Length of main landing gear (in)	(Main Landing Gear)
25	0	CGmgr	x-position of the main landing gear c.g. (ft)	
26	0	Ln	Length of nose gear (in)	(Nose Gear)
27	0	CGngr	x-position of the nose gear c.g. (ft)	
28	1	Pmax	Maximum power at sea level (hp)	(Power)
29	1	Wguess	Guess initial weight for sizing (lb)	(Weights)
30	0	XM	Distance of c.g from the nose (ft)	

***Step #4**: Define the lower and upper bounds on the design variables using* **lower.wk1** *and* **upper.wk1.**

Looking at these files, the user will notice that the setup of these files is identical to the screen-shot shown for **variables.wk1**. Using these files, the user must set the lower and upper bounds on the used design variables using the respective input file.

***Step #5**: Input values for the user-defined constraints using* **UserConstraints.wk1**

As discussed in the body of the thesis, the user can establish restrictions on a number of aircraft stability and control parameters. These values are then used to determine whether a constraint is violated or met. The input screen appears as:



1	3	SMmax	Maximum allowable static margin
2	0.02	SMmin	Minimum allowable static margin
3	-1	CYBo	Limit on the stability derivative, Cybeta [-]
4	-0.2	CLBo	Limit on the stability derivative, Clbeta (-)
5	0.2	CNBo	Limit on the stability derivative, Cnbeta (+)
6	20	DEmax	Maximum positive elevator deflection (deg)
7	20	DEmin	Maximum negative elevator deflection (deg)
8	20	DAmax	Maximum positive aileron deflection (deg)
9	20	DAmin	Maximum negative aileron deflection (deg)
10	20	DRmax	Maximum positive rudder deflection (deg)
11	20	DRmin	Maximum negative rudder deflection (deg)
12	0.04	PhDmin	Minimum value for the Phugoid damping ratio
13	0.3	SPmin	Minimum value for the Short Period damping ratio
14	2	SPmax	Maximum value for the Short Period damping ratio
15	3.6	SPnatmax	Maximum value for the Short Period natural frequency
16	0.28	SPnatmin	Minimum value for the Short Period natural frequency
17	0.08	DRdmin	Minimum value for the Dutch Roll damping ratio
18	1.4	Trollmax	Maximum roll mode time constant (sec)
19			

Step #6: Using **ConstraintsUsed.wk1** *and* **EQconsUsed.wk1,** ***define which inequlity and equality constraints are to be used.***

These two files are setup very similar to the file, **usedvars.wk1**. In each, a complete list of possible constraints is listed. A "1" must be placed next to each constraint that is to be used. A "0" indicates that the constraint will not be used. The next section discusses how to modify **ConstraintsUsed.wk1** to support a variety of mission definitions.

Step #7: Save the input files

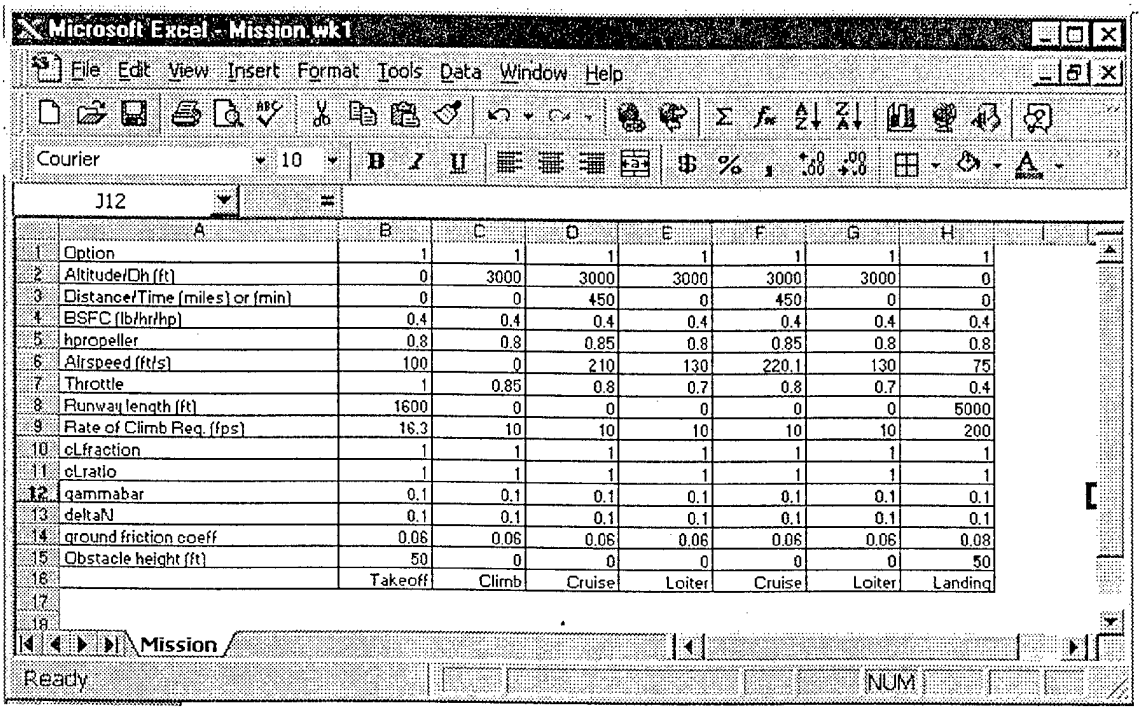
To clarify for the first time user, saving instructions have been added here to avoid confusion. When saving, make sure that the selected file type is "WK1(1-2-3) (*.wk1)" and simply overwrite using the same file name. Due to the format of these files, *Excel* will ask if you want to convert them to **.xls* format upon closing. Do not do this. Select "No" and move on.

B.3 Defining the mission profile

Defining the mission profile is the most intensive task facing the user. This is the penalty of allowing any number of various mission definitions. With this in mind, proceed with caution.

***Step #8**: Define the mission using* **mission.wk1**

This file consists of a number of columns, each column representing one mission leg, which must be built and values added, to completely define the mission profile. The following screen shot shows a mission profile consisting of seven mission legs.

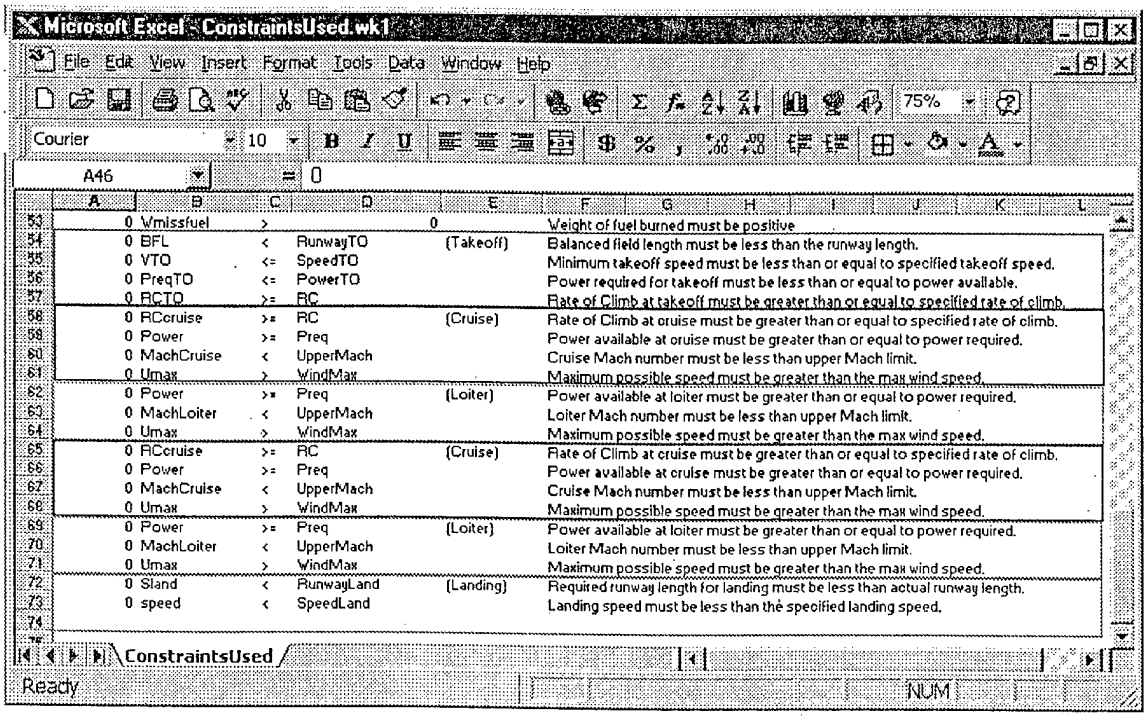


	A	B	C	D	E	F	G	H
1	Option	1	1	1	1	1	1	1
2	Altitude/Dh (ft)	0	3000	3000	3000	3000	3000	0
3	Distance/Time (miles) or (min)	0	0	450	0	450	0	0
4	BSFC (lb/hr/hp)	0.4	0.4	0.4	0.4	0.4	0.4	0.4
5	hpropeller	0.8	0.8	0.85	0.8	0.85	0.8	0.8
6	Airspeed (ft/s)	100	0	210	130	220.1	130	75
7	Throttle	1	0.85	0.8	0.7	0.8	0.7	0.4
8	Runway length (ft)	1600	0	0	0	0	0	5000
9	Rate of Climb Req. (fps)	16.3	10	10	10	10	10	200
10	cL fraction	1	1	1	1	1	1	1
11	cL ratio	1	1	1	1	1	1	1
12	gammabar	0.1	0.1	0.1	0.1	0.1	0.1	0.1
13	deltaN	0.1	0.1	0.1	0.1	0.1	0.1	0.1
14	ground friction coeff	0.06	0.06	0.06	0.06	0.06	0.06	0.08
15	Obstacle height (ft)	50	0	0	0	0	0	50
16		Takeoff	Climb	Cruise	Loiter	Cruise	Loiter	Landing
17								
18								

These columns must always be filled from left to right, beginning with column B. Adding a new leg simply requires inserting a column at the desired position. If removing an existing leg, simply delete and shift the remaining legs to the left-most position.

***Step #9**: Modify* **ConstraintsUsed.wk1** *to reflect the new mission profile*

Adding or removing mission legs directly effects the number of entries into the constraint vector. For this reason, the **ConstraintsUsed.wk1** file must be modified to reflect the addition or removal of mission constraints. The following figure shows the end portion of the **ConstraintsUsed.wk1** file. The mission constraints begin at position 54, as shown. This should never change. Each mission leg has a particular set of associated constraints (refer to Table 10.4.6), which have been highlighted in the following screen shot.

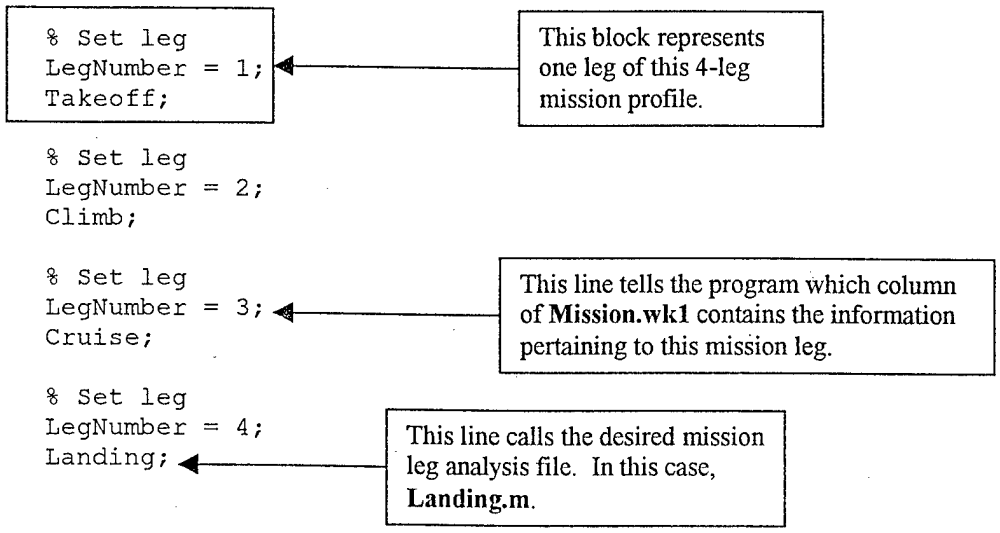


53	0	Wmissfuel	>	0		Weight of fuel burned must be positive
54	0	BFL	<	RunwayTO	(Takeoff)	Balanced field length must be less than the runway length.
55	0	VTO	<=	SpeedTO		Minimum takeoff speed must be less than or equal to specified takeoff speed.
56	0	PreqTO	<=	PowerTO		Power required for takeoff must be less than or equal to power available.
57	0	RCTO	>=	RC		Rate of Climb at takeoff must be greater than or equal to specified rate of climb.
58	0	RCcruise	>=	RC	(Cruise)	Rate of Climb at cruise must be greater than or equal to specified rate of climb.
59	0	Power	>=	Preq		Power available at cruise must be greater than or equal to power required.
60	0	MachCruise	<	UpperMach		Cruise Mach number must be less than upper Mach limit.
61	0	Umax	>	WindMax		Maximum possible speed must be greater than the max wind speed.
62	0	Power	>=	Preq	(Loiter)	Power available at loiter must be greater than or equal to power required.
63	0	MachLoiter	<	UpperMach		Loiter Mach number must be less than upper Mach limit.
64	0	Umax	>	WindMax		Maximum possible speed must be greater than the max wind speed.
65	0	RCcruise	>=	RC	(Cruise)	Rate of Climb at cruise must be greater than or equal to specified rate of climb.
66	0	Power	>=	Preq		Power available at cruise must be greater than or equal to power required.
67	0	MachCruise	<	UpperMach		Cruise Mach number must be less than upper Mach limit.
68	0	Umax	>	WindMax		Maximum possible speed must be greater than the max wind speed.
69	0	Power	>=	Preq	(Loiter)	Power available at loiter must be greater than or equal to power required.
70	0	MachLoiter	<	UpperMach		Loiter Mach number must be less than upper Mach limit.
71	0	Umax	>	WindMax		Maximum possible speed must be greater than the max wind speed.
72	0	Sland	<	RunwayLand	(Landing)	Required runway length for landing must be less than actual runway length.
73	0	speed	<	SpeedLand		Landing speed must be less than the specified landing speed.
74						

For example, if the second cruise leg were removed, the constraints following would have to be shifted up resulting in only 71 possible constraints instead of 74.

Step #10*: *Modify* Mission.m *to reflect the new mission profile

The source code for **Mission.m** is included in Appendix C and should be referenced during this discussion. This program file is not automatically updated when the user modifies **Mission.wk1** and therefore must be modified to reflect the new mission profile. The following is an example of how to program a mission profile consisting of takeoff, climb, cruise, and landing.



As long as the above example is followed, the user can define any number of mission legs for analysis. This step completes the task of defining the mission profile.

B.4 Running the program

***Step # 11*: *Running* OPT1st.m *and* analyze.m**

Running the program is exceptionally simple. First, open *MATLAB* and access the directory in which all your files are located. The user now has two options:

1.) Type **OPT1st** and hit return. This launches the optimization program and will run until all the constraints have been satisfied. The "optimized" vector of design variables will appear on the screen.

2.) Type **analyze** and hit return. This launches the analysis program only and will perform all possible analysis on the design which is currently defined in the **variables.wk1** and **parameters.wk1** files.

** Warning...A couple things to be aware of:

1.) During the course of optimization the program will sometimes violate restrictions within the Smetana codes. The Smetana code responds by printing a violation message to the screen. The user may or may not observe such messages. These messages do not effect the course of optimization in any way.

2.) Changes to the input files sometimes cause a vector dimensions error at the *MATLAB* prompt if these files remain open. It is unknown as to the source of this problem. The best way around this is to re-save the input files, type **clear** at the *MATLAB* prompt, and try to run the program again. Be persistent, it will work.

B.5 Interpreting the results

Now that you have successfully defined and run your design problem, it is time to view the all-important results. These results are contained in four output files that are automatically created:

1.) LongStabOut.txt: This file contains a listing of the aircraft geometric parameters and the static and dynamic longitudinal stability results.

2.) LatDirOut.txt: This file presents all of the static and dynamic lateral-directional stability results.

3.) MissionOutput.txt: This file contains the results of each mission leg and also the fuel fractions for each leg.

4.) Output.txt: This file contains a variety of results to include control surface deflection, ground handling stability, and the component weight breakdown estimation. Also included is a complete list of the constraints and the design variables.

Another way that was developed to view the resulting design is the **plotdesign.m** graphical design program. To utilize this program, the resulting design variables must be entered into the **variables.wk1** file. The user must then run **analyze.m** to calculate the necessary internal program parameters. Once this has been done, type **plotdesign** at the *MATLAB* prompt. Two figures will appear. One, which depicts the top view of your design, and one, which shows the side view. Examples of this graphical output are presented in chapter 12.

APPENDIX C

SOURCE CODE LISTING

C.1 Introduction

The pages contained in this appendix present the total source code listing for each module. A short description precedes each listing to provide an understanding of the specific function performed.

C.2 OPT1st.m

This is the optimization module "executable" file. When typed at the *MATLAB* window, it performs the design case defined by the user.

```
% OPT1st.m: Optimization Design Executable
% Calling statements for Optimizer
% Global statements allow these variables to be accessed outside of
% the MATLAB function environment
clear
global g
global geq
global DV
global X
global PP
global C
global Ceq
global EQconsUsed

%Design Variables
□

DV = wk1read('variables.wk1');
□
%Lower Bounds on Design Variables
□
XLB = wk1read('lower.wk1');
□
%Upper Bounds on Design Variables
□
XUB = wk1read('upper.wk1');
□
Xused = wk1read('usedvars.wk1');
```

```
% Write the vector of used design variables
j = 1;
for i = 1:1:length(DV)
   if Xused(i) == 1
        X0(j) = DV(i);
        LBo(j) = XLB(i);
        UBo(j) = XUB(i);
        j = j+1;
   end
end
X0 = transpose(X0);
LBo = transpose(LBo);
UBo = transpose(UBo);
%<---- used to initialize interface
%X = X0;%X0 <---- Input Design Variables

% Matrices available for linear optimization problems
Aoo = [];
Boo= [];
Aeqo = [];
Beqo = [];
OPTIONS = optimset('TolX',0.00000000000000000001,'MaxFunEvals',500);

% Initiates the FMINCON function using the user inputs.
% This command is the execution of the optimizer.
[X,FVAL,EXITFLAG,OUTPUT] =
fmincon('ANALfun1st',X0,Aoo,Boo,Aeqo,Beqo,LBo,UBo,'NONLCON1st',OPTIONS);

disp(X);
```

C.3 ANALFUN1st.m

The *FMINCON MATLAB* function requires the analysis module to be executed in the form of a function, rather than a standard *.m* file. Thus, this function file calls the analysis module executable file, *analyze1st.m*.

```
% ANALFUN1st.m
% This file is defined as a function file using the following line
% The objective function, F, and the design variables, X, must be
% formatted in this manner to work with FMINCON.

function [F] = ANALfun1st(X)

global g
global geq
global DV
global Xused
global PP

%Open input file
format long
%Permanent Parameters
PP = wk1read('parameters.wk1');
j = 1;
```

```
DV = wk1read('variables.wk1');
Xused = wk1read('usedvars.wk1');


% Write the X vector of design variables into the list of unused design variables, DV.
for i = 1:1:length(DV)
    if Xused(i) == 1
        DV(i,1) = X(j);
        j = j+1;
    end
end

% Call analyzelst.m, the analysis module
analyzelst;

% Define Objective Function
F = Wcalc;
```

C.4 NONLCON1st.m

This file is also a function to fit with the formatting requirements of *FMINCON*. It takes the complete constraint vectors, *g* and *geq*, and maps them to the vectors of used constraints, *C* and *Ceq*.

```
% NONLCON1st.m
% Defines the vectors of used constraints

function [C,Ceq] = NONLCON1st(X,F)

% Dummy constraint code used to grab constraints from
% analysis code.

global g
global geq
global C
global Ceq
% Ensures these vectors are written as column vectors
C = -ones(3,1);
Ceq = zeros(3,1);

% Map constraints
% Read user input files
CONSused = wk1read('ConstraintsUsed.wk1');
EQconsUsed = wk1read('EQconsUsed.wk1');
j = 1;
for i = 1:1:length(g)
   if CONSused(i,1) == 1
        C(j) = g(i);
        j = j+1;
   end
end
j = 1;
```

```
for i = 1:1:length(geq)
   if EQconsUsed(i,1) == 1
        Ceq(j,1) = geq(i);
        j=j+1;
   end
end
```

C.5 analyze1st.m

This file is the analysis module executable that has been modified from *analyze.m* to allow integration with the optimization module. The variables used within this file are first defined in *ANALFUN1st.m*.

```
%analyze1st.m
%Called from within ANALFUN1st.m, this launches the analysis module
%Order is important
□
%*****************************************************************************
□
cnn = 0;
□

□
%Call Geometry.m to calculate design configuration
%Defines all geometric parameters utilized in all other files
Geometry;

%Call Mission.m to perform initial sizing and mission analysis
%Airspeed at cruise used in determination of stability derivatives
%If this section is omitted, airspeed must be defined elsewhere
□
Mission;

%Call Weights.m to calculate weights and c.g.
%Moments of inertia used for dynamic stability analysis are determined here
□
Weights;
□

□
%Call LandingGear.m to develop landing gear constraints
LandingGear;
□

□
%Call VdocLongStab.m to calculate Longitudinal Stability Derivatives
%for the current design
VdocLongStab;
□

□
%Call VdocLongDynm.m to calculate Longitudinal Dynamic Response
□
%for the current design
□
```

```
VdocLongDynm;
□

□
%Call VdocLatDir.m to calculate Lateral-directional Stability Derivatives
□
%for the current design
VdocLatDir;

%Call VdocLatDynm.m to calculate Lateral-directional Dynamic Response
%for the current design
VdocLatDynm;

%Call Controls.m to determine control-surface deflection
Controls;

%Call Allconstraints.m to calculate constraints and generate vectors
Allconstraints;
```

C.6 analyze.m

There is very little difference between this file and *analyze1st.m* except that this file can be executed as a stand-alone file. Basically, if the user simply wants to analyze a specific design, that design can be expressed in the input files and the analysis performed by simply using this file.

```
%analyze.m
□
%stand-alone analysis executable file
□
%performs the complete analysis on a user defined design
%no optimization involved.
□
%*****************************************************************************

%Open input file
format long
%Permanent Parameters
PP = wk1read('parameters.wk1');
%Design Variables
DV = wk1read('variables.wk1');
X = DV;
%Lower Bounds on Design Variables
XL = wk1read('lower.wk1');
%Upper Bounds on Design Variables
XU = wk1read('upper.wk1');

%Call Geometry.m to calculate design configuration
%Defines all geometric parameters utilized in all other files
□
Geometry;
□
```

```
□
%Call Mission.m to perform initial sizing and mission analysis
%Airspeed at cruise used in determination of stability derivatives
%If this section is omitted, airspeed must be defined elsewhere
□
Mission;
□

□
%Call Weights.m to calculate weights and c.g.
%Moments of inertia used for dynamic stability analysis are determined here
Weights;

%Call LandingGear.m to develop landing gear constraints
LandingGear;
□

□
%Call VdocLongStab.m to calculate Longitudinal Stability Derivatives
□
%for the current design
VdocLongStab;
□

□
%Call VdocLongDynm.m to calculate Longitudinal Dynamic Response
%for the current design
VdocLongDynm;

%Call VdocLatDir.m to calculate Lateral-directional Stability Derivatives
%for the current design
VdocLatDir;

%Call VdocLatDynm.m to calculate Lateral-directional Dynamic Response
%for the current design
VdocLatDynm;

%Call Controls.m to determine control-surface deflection
Controls;

%Call Allconstraints.m to calculate constraints and generate vectors
Allconstraints;
```

C.7 Geometry.m

This file defines and calculates all the required geometric information pertaining to the design.

```
% Program Geometry.m
□
% This program consolidates all geometry/configuration calculations, originally
□
% performed within the Smetana codes, into this one code.
□
%*****************************************************************************
```

```
% Read Flight Conditions
GAMMA = PP(83); GRAVIT = PP(88);

% Convert GAMMA to radians and find g*cos(gamma) and g*sin(gamma)
GAMMA = GAMMA*pi/180;
GCOSGM = cos(GAMMA);
GSINGM = sin(GAMMA);

% Wing data (geometry)
B = DV(1);
 CROOT = DV(2); TR = DV(3); IWING = PP(28);
□
SA = PP(27); DIH = DV(4); ZW = PP(81); Xlewng = DV(5);
□
toverc = PP(29); tc = toverc;
□

□
% Added as safeguards against negative geometric distances
% Appear to assist the optimizer in most situations
if B <= 0
   B = 10;
end
if TR > 1
   TR = 1;
end
if TR < 0.5
   TR = 0.5;
end
if CROOT <= 0
  CROOT = 2;
end

% Cbar (average chord for area calculation)
□
Cbar = 0.5*(CROOT+TR*CROOT);
□
MACwing = (2/3)*CROOT*(1+TR+TR^2)/(1+TR); %Raymer pg. 49

% Calculate wing area
S = B*Cbar;
CROOT = Cbar;

% Wing Data (aerodynamic)
CLA2DW = PP(42); CDA2DW = PP(37);
ALPHA = PP(82);

% Convert ALPHA to radians
ALPHA = ALPHA*pi/180;

% Horizontal Tail Data (geometry/location)
BT = DV(6); Crtail = DV(7); TRT = DV(8); Xletail = DV(9);
ITAIL = PP(31); ZT = PP(79); EFF = PP(47);
SAH = PP(30); BH = BT; TRH = TRT;

% Added as safeguards against negative geometric distances
% Appear to assist the optimizer in most situations
if TRT > 1
   TRT = 1;
end
if TRT < 0.5
   TRT = 0.5;
end
```

```
if BT <= 0
  BT = 2;
end

%Cbartail (average chord for area calculation)
□
Cbartail = 0.5*(Crtail + TRT*Crtail);
□
MACht = (2/3)*Crtail*(1+TRT+TRT^2)/(1+TRT); %Raymer pg. 49
□

□
%Calculate horizontal tail area
□
ST = BT*Cbartail;
SH = BH*Cbartail;

% Horizontal Tail Data (Aerodynamic)
CLA2DT = PP(41);
CLA2DH = CLA2DT;

% Vertical Tail data
BV = DV(10);  Crvt = DV(11); TRvt = DV(12); Xlevt = DV(13);
ZV = PP(80); ETAV = PP(48); SAvt = PP(32);

% Added as safeguards against negative geometric distances
% Appear to assist the optimizer in most situations
if BV <=0
  BV = 2;
end
if Crvt <=0
  Crvt = 1;
end

Cbarvt = .5*(Crvt + TRvt*Crvt);
□
MACvt = (2/3)*Crvt*(1+TRvt+TRvt^2)/(1+TRvt); %Raymer pg. 49
□
SV = BV*Cbarvt;
XVT = Xlevt + .25*Crvt;

% Aspect Ratios of wing and tails.
AR = B^2/S;
ART = BT^2/ST;

% Elevator Data
CHE = DV(20); BE = .5*BT;

% Added as safeguards against negative geometric distances
% Appear to assist the optimizer in most situations
if BE <=0
  BE = 0.5;
end
if CHE <=0
  CHE = 0.2;
end

% Calculate elevator area assuming rectangular   Account for both sides of tail
SELEV = 2*CHE*BE;

□
```

```
% Aileron Data
□
BA = DV(14); CA = DV(15); YI = DV(16);
□

% Added as safeguards against negative geometric distances
% Appear to assist the optimizer in most situations
□
if BA <=0
  BA = 0.5;
end
if CA<=0
  CA = 0.2;
end
if YI <=0
  YI = 2;
end

SAileron = BA*CA;
% Flap Data
□
BFlap = DV(17);
□
CFlap = DV(18);
□

% Added as safeguards against negative geometric distances
% Appear to assist the optimizer in most situations
□
if BFlap <=0
  BFlap = 0.5;
end
if CFlap <=0
  CFlap = 0.2;
end

% Rudder Data
□
BR = BV; CR = DV(22);
□
if BR <= 0
  BR = 1;
end
if CR <=0
  CR = 0.2;
end
SR = BR*CR;
□

□
% Fuselage Initialized Data (Geometry)
LB = DV(23); Lo = PP(1); WFUSo = PP(13);
H1o = PP(2); H2o = PP(4); Ho = PP(2); R1o = PP(12);
LF = LB; HNOSEo = PP(8); WNOSEo = PP(16); HFCYo = PP(6); WFCYo = PP(15);
LFCYo = PP(10); LMHo = PP(11); HBCYo = PP(5); WBCYo = PP(14); LBCYo = PP(9);
Hgearo = PP(7); Option = PP(26);

% Fuselage Data (aerodynamic)
CLAFUS = PP(43);
LDmax = PP(50);

% Configuration data (geometric)
ZA = PP(78); XM = DV(30);
```

```
□

□
% Calculate important distances
□
XA = XM - (Xlewng + .25*CROOT);
□
XFUS = Xlewng + .25*CROOT;
XCG = -XA;
LT = (Xletail + .25*Crtail) - XM;
LT1 =  (Xletail + .25*Crtail) - (Xlewng + .25*CROOT);
XW = XFUS;
XHT = Xletail + .25*Cbartail;

% Configuration data (aerodynamic)
CD0 = PP(35);
cD0 = CD0;

% Thrust data
ZJ = ZT; COSXZ = PP(84); SINXZ = PP(85);

% Geometry for weights calculations
HtHv = PP(68); Lm = DV(24); Ln = DV(26);
Nen = PP(54); Nl = PP(69); Nt = PP(72); Nz = PP(73);
Pdelta = PP(74); Vpr = PP(75); Wguess = DV(29); Wdg = Wguess;
Wen = PP(55); Wl = PP(76); Wuav = PP(63); Nstrut = PP(70);
Npassengers = PP(57); Ncrew = PP(58); Wppass = PP(59);
Wpcrew = PP(60);
Np = Npassengers+Ncrew;
% Total Person Weight
Wcrew = Wpcrew*Ncrew+Wppass*Npassengers;

□
% CG locations of various aircraft systems and components
CGwing = Xlewng +.4*MACwing;
CGht = Xletail + .4*MACht;
CGvt = Xlevt + .4*MACvt;
CGfuse = .5*LB;
CGmgr = DV(25);
CGngr = DV(27);
CGengo = PP(21); CGfuelo = PP(22);
□
CGctrlo = PP(19); CGhydro = PP(24); CGeleco = PP(20); CGaviono = PP(18);
□
CGaco = PP(17); CGfurno = PP(23);

% Convert sweep angles to radians
SA = SA*pi/180;
SAH = SAH*pi/180;
SAvt = SAvt*pi/180;

% Add permanent parameters and design variables required for performance
% analysis within the mission profiles
AnPower = PP(52); cDFlapRatio = PP(38); cDLGratio = PP(39);
cFriction = PP(40); cLmax = PP(44); cLmaxClean = PP(45);
Dprop = PP(56); EnginDragFactor = PP(46); engineDref = PP(33);
engineLref = PP(34); enginePowerref = PP(53); eps = PP(77);
groundeffect = PP(49); Growth = PP(62);
Oswald = PP(51); RoskamA = PP(64);
RoskamB = PP(65); slender = PP(25);
UpperMach = PP(86); WindMax = PP(87); Wpayload = PP(61);
Pmax = DV(28); RaymerA = PP(66); RaymerC = PP(67);
```

```
% Call Fuselage.m to determine fuselage dimensions "Rubber Fuselage Concept"
Fuselage;

%Find approximate fuselage wetted area
Atop = LB*(WNOSE+WFCY+WBCY+WFUS)/4;
Aside = LB*(HNOSE+HFCY+HBCY)/3;
Sf = 3.4*(Atop+Aside)/2;  %Eqn. 7.12 in Raymer

%Determine wing wetted area
Sexposed = S-CROOT*WFUS;
Swingwet = Sexposed*(1.977+0.52*toverc);

%Horizontal tail and vertical tail wetted areas are just 2 times normal
Swet = Sf+Swingwet+2*ST+2*SV;

□
% Call Parasite.m to determine parasite drag coefficient
□
Parasite;
□

□
% Determine total tank volume for integral fuel tanks
% From Torenbeek pg. 449 eqn. B-12
Vi = 0.54*S^2*toverc/B*(1+TR+TR^2)/(1+TR)^2; %ft^3
% Convert to gallons and multiply by Mil-Spec density of 6.0 lb/gal
Vi = 7.481*Vi;
Vt = Vi;
Fueldens = PP(71);
Wfw = Vi*7.481*Fueldens;
```

C.8 Fuselage.m

This file establishes the fuselage dimensions based on one of three options, which must be set by the user. The file itself is called from within *Geometry.m*.

```
%Fuselage.m
□
%This code determines the required fuselage dimensions based on the %input values and the
option selected by the user.
%Option 1: Use the given values
%Option 2: Scale the fuselage to the length
%Option 3: Using the slenderness ratio, scale the fuselage to the width

if Option == 1
   % Set fuselage dimensions to input values
   % LB is initialized as a design variable and is modified by optimizer
   WFUS = WFUSo; H = Ho; H1 = H1o; H2 = H2o; HBCY = HBCYo; HFCY = HFCYo;
   Hgear = Hgearo; HNOSE = HNOSEo; LBCY = LBCYo; LFCY = LFCYo; LMH = LMHo;
   R1 = R1o; WBCY = WBCYo; WFCY = WFCYo; WNOSE = WNOSEo; CGac = CGaco;
   CGavion = CGaviono; CGctrl = CGctrlo; CGelec = CGeleco; CGeng = CGengo;
   CGfuel = CGfuelo; CGfurn = CGfurno; CGhydr = CGhydro;

elseif Option == 2
   % Scale fuselage dimensions using length ratios
```

```
   WFUS = WFUSo/Lo*LB; H = Ho/Lo*LB; H1 = H1o/Lo*LB; H2 = H2o/Lo*LB;
   HBCY = HBCYo/Lo*LB; HFCY = HFCYo/Lo*LB; Hgear = Hgearo/Lo*LB;
   HNOSE = HNOSEo/Lo*LB; LBCY = LBCYo/Lo*LB; LFCY = LFCYo/Lo*LB;
   LMH = LMHo/Lo*LB; R1 = R1o/Lo*LB; WBCY = WBCYo/Lo*LB; WFCY = WFCYo/Lo*LB;
   WNOSE = WNOSEo/Lo*LB; CGac = CGaco/Lo*LB; CGavion = CGaviono/Lo*LB;
   CGctrl = CGctrlo/Lo*LB; CGelec = CGeleco/Lo*LB; CGeng = CGengo/Lo*LB;
   CGfuel = CGfuelo/Lo*LB; CGfurn = CGfurno/Lo*LB; CGhydr = CGhydro/Lo*LB;

else
   % Scale fuselage dimensions using fixed slenderness ratio
       WFUS = 1/slender*LB; H = Ho/WFUSo*WFUS; H1 = H1o/WFUSo*WFUS; H2 = H2o/WFUSo*WFUS;
   HBCY = HBCYo/WFUSo*WFUS; HFCY = HFCYo/WFUSo*WFUS; Hgear = Hgearo/WFUSo*WFUS;
   HNOSE = HNOSEo/WFUSo*WFUS; LBCY = LBCYo/WFUSo*WFUS; LFCY = LFCYo/WFUSo*WFUS;
   LMH = LMHo/WFUSo*WFUS; R1 = R1o/WFUSo*WFUS; WBCY = WBCYo/WFUSo*WFUS;
   WFCY = WFCYo/WFUSo*WFUS; WNOSE = WNOSEo/WFUSo*WFUS; CGac = CGaco/WFUSo*WFUS;
   CGavion = CGaviono/WFUSo*WFUS; CGctrl = CGctrlo/WFUSo*WFUS;
   CGelec = CGeleco/WFUSo*WFUS; CGeng = CGengo/WFUSo*WFUS; CGfuel = CGfuelo/WFUSo*WFUS;
   CGfurn = CGfurno/WFUSo*WFUS; CGhydr = CGhydro/WFUSo*WFUS;
end

YFlap = WFUS/2;
```

C.9 Parasite.m

This code determines the parasite drag coefficient if desired by the user. It is called from *Geometry.m*.

```
% Parasite.m
%
%
% This code determines the value for the parasite drag coefficient based on
%
% the user's desired method
%
%
% If CD0 == 0 then the code calculates CD0 using Torenbeek's wetted area
% approximation and the aerodynamic cleanness coefficient

if CD0 == 0

   %Find approximate fuselage wetted area
       Atop = LB*(WNOSE+WFCY+WBCY+WFUS)/4;
       Aside = LB*(HNOSE+HFCY+HBCY)/3;
       Sf = 3.4*(Atop+Aside)/2;  %Eqn. 7.12 in Raymer

       %Determine wing wetted area
       Sexposed = S-CROOT*WFUS;
       Swingwet = Sexposed*(1.977+0.52*toverc);

       %Horizontal tail and vertical tail wetted areas are just 2 times normal
       Swet = Sf+Swingwet+2*ST+2*SV;

       CD0 = Swet/Swing*CF;
end
```

C.10 Mission.m

This file reads in the mission definition and organizes and conducts the total mission analysis. New mission definitions must be indicated by changes in this file and in the input file.

```
% Mission.m

% User modifiable program to perform mission analysis utilizing any combination of

% prescribed mission legs: Takeoff, Climb, Cruise, Loiter, & Landing. Descent
% concerns are included in the landing leg calculations.

% This code calls the file, mission.wk1, as input for each leg.
% The variable LegNumber must be set between each leg based on order.

% Do not modify
% This allows mission analysis beginning with any leg...does not necessarily
% have to be a takeoff leg.
% Remember to input Wguess into main input file
Win = Wguess;

% Open input file
MissIN = wk1read('Mission.wk1');
gmission = [];
% Generate sea level reference conditions
□
altitude = 0;
□
Atmos;
□
densSL = rho;
□

□
% Create mission output file for constraint values
fid4 = fopen('MissionOutput.txt','w');
fprintf(fid4,'\n Mission Performance Results');
fprintf(fid4,'\n');

%%%% USER MODIFIABLE SECTION!!! %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Following setup is for a mission consisting of six mission legs
% Follow the same formatting when adding mission legs

□
% Set leg
LegNumber = 1;
Takeoff;

% Set leg
LegNumber = 2;
Climb;

% Set leg
LegNumber = 3;
Cruise;
```

```
% Set leg
□
LegNumber = 4;
□
Loiter;
□

□
% Set leg
□
LegNumber = 5;
□
Cruise;
□

□
% Set leg
□
LegNumber = 6;
□
Loiter;
□

□
% Set leg
□
LegNumber = 7;
□
Landing;

% USER MODIFIABLE REGION ENDS HERE!!
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Final weight
□
Wfinal = Win;
□
% Set objective as minimizing the final weight
□
OBJ = Wfinal;
□

□
% Fuel weight
□
WmissFuel = Wguess-Wfinal;

% Any number of mission legs can be entered in the above format.  The variable
% LegNumber, indicates the column in mission.wk1 that contains the particular
% definitions for each leg.

%Empty Weight Calculation
We1 = Wguess-WmissFuel-Wpayload-Wcrew;
Wroskam = 10^((log10(Wguess)-RoskamA)/RoskamB);
Wraymer = Wguess*RaymerA*Wguess^RaymerC;
We2 = 0.937*(Wroskam+Wraymer)/2;

%Gross Weight Calculation
Wcalc = (Wcrew+Wpayload)/(1-(WmissFuel/Wguess)-(We2/Wguess)); %Raymer
%Wcalc = We2+WmissFuel+Wpayload+Wcrew; %Used by Eli
gWeight = (Wcalc - Wguess)/Wguess;

fprintf(fid4,'\n');
```

```
fprintf(fid4,'\n Mission Segment Fuel Fractions:');
fprintf(fid4,'\n %4.6f',WratiosA);
```

C.11 Takeoff.m

This file performs the analysis for the takeoff mission leg. It is called from *Mission.m*.

```
% Takeoff.m
□
% This file performs all aspects of the takeoff analysis
□

□
fprintf(fid4,'\n Takeoff:');
fprintf(fid4,'\n');

% Read input pertinent to the takeoff leg
IN = MissIN(:,LegNumber+1);
□
option = IN(1);
□
altitude = IN(2);
Np = IN(5);
SpeedTO = IN(6);
throttle = IN(7);
RunwayTO = IN(8);
RCTO = IN(9);
groundFriction = IN(14);
Hobstacle = IN(15);
□
Atmos;
□

□
% Performance method (Balanced field length)
□
sigma = rho/densSL;
□
Power = Pmax*(sigma)^AnPower; %hp
□
cLclimb = cLmax/1.44;
Ur = 0.01*cLmax+0.2;
G = 0; % Set as such for preliminary sizing

% Average thrust from Raymer, pg. 489
Taverage = 5.75*Power*(sigma*Nen*Dprop^2/Power)^(1/3);

% Balanced Field Length from Raymer, pg. 489
BFL = 0.863/(1+2.3*G)*((Win/S)/(rho*GRAVIT*cLclimb)+Hobstacle)*(1/(Taverage/Win ...
      -Ur)+2.7)+(655/sqrt(sigma));

% Takeoff CL, CD, Airspeed, Power Required
cLTO = cLmax/1.44;
VTO = sqrt(2*Win/rho/S/cLTO);
ReTO = rho*VTO*CROOT/visc; %Reynold's number at Takeoff
cDTO = cD0+cLTO^2/pi/AR/S*groundeffect+cDLGratio*cD0+cDFlapRatio*cD0;
DragTO = 0.5*rho*VTO^2*S*cDTO; %lb
PreqTO = DragTO*VTO; %ft-lb/s
PreqTO = PreqTO/550; %hp
PowerTO = Np*Pmax*sigma^AnPower; %hp
```

```
% Rate of climb at takeoff
PowerClimb = Power*throttle; %hp
AA = Np*PowerClimb*550/Win;
BB = 0.5*rho*S*cD0/Win;
CC = Win/pi/AR/Oswald*2/rho/S;
% Optimum Climb speed
Uopt = sqrt(sqrt(CC/3/BB));
% Optimum Rate of Climb
RC = AA-BB*Uopt^3-CC/Uopt;

% BFL must be less than the runway length
gTO(1,1) = abs(BFL/RunwayTO)-1;
% VTO must be less than or equal to specified Takeoff speed
gTO(2,1) = VTO/SpeedTO-1;
% Power available must be greater than or equal to power required
gTO(3,1) = PreqTO/PowerTO-1;
% Rate of Climb at takeoff greater than or equal to desired rate of climb
gTO(4,1) = 1 - RC/RCTO;

% Statistical analysis
% warmup and take-off (Raymer Table 3.2)
Wout= Win*0.97;

WratiosA(LegNumber,1)=Wout/Win;
Win = Wout;

gmission = [gmission; gTO];
fprintf(fid4,'\n Balanced Field Length, ft         = %4.6f',BFL);
fprintf(fid4,'\n Takeoff Airspeed, ft/s            = %4.6f',VTO);
fprintf(fid4,'\n Takeoff Lift Coefficient          = %4.6f',cLTO);
fprintf(fid4,'\n Takeoff Drag Coefficient          = %4.6f',cDTO);
fprintf(fid4,'\n Power Available, hp               = %4.6f',PowerTO);
fprintf(fid4,'\n Power Required, hp                = %4.6f',PreqTO);
fprintf(fid4,'\n Rate of Climb, ft/s               = %4.6f',RC);
```

C.12 Climb.m

This file outlines the climb analysis and calls the *BestClimb.m* routine. *Climb.m* is called by *Mission.m*.

```
% Climb.m
% Perform climb analysis

fprintf(fid4,'\n');
fprintf(fid4,'\n Climb:');
fprintf(fid4,'\n');

IN = MissIN(:,LegNumber+1);
option = IN(1);
deltah = IN(2);
BSFC = IN(4);
Np = IN(5);
throttle = IN(7);

% Call BestClimb.m
Eliclimb;
```

```
if option == 1
   % Statistical analysis
   % warmup and take-off (Raymer Table 3.2)

        Wout= Win*0.985;
end

WratiosA(LegNumber,1)=Wout/Win;
Win = Wout;
fprintf(fid4,'\n Optimum Climb Speed, ft/s          = %4.6f',Uclimb);
fprintf(fid4,'\n Optimum Lift Coefficient           = %4.6f',cLstar);
fprintf(fid4,'\n Climb Drag Coefficient             = %4.6f',cD);
fprintf(fid4,'\n Power Available, hp                = %4.6f',P/550);
fprintf(fid4,'\n Rate of Climb, ft/s                = %4.6f',dhdt);
```

C.13 BestClimb.m

This file performs the analysis for optimum climb.

```
% BestClimb.m (ELiClimb.m)
□
% Best climb routine
□

□
AMmax = UpperMach;
% find opt cL for max cL/cD
□
cLoptcLcD=sqrt(pi*AR*Oswald*cD0);
□

□
% corrected opt cL/cD
□
if cLoptcLcD > cLmax/1.4
   cLoptcLcD = cLmax/1.4;
end

% h increment
nsegments = 10;
dh = deltah/nsegments;

% aerodynamic drag cD=cD0+AK*cL**2
AK=1/pi/AR/Oswald;

% before start of numerical integration set initial values
h=altitude-dh;
W=Win;
distance=0;
time=0;

% initial gamma (rad)
gamma=0;

for i=1:1:nsegments

   h=h+dh;
   Atmos;
```

```
     % find Ustar (optimum climb speed, Raymer p. 458, eq. 17.19)
UstarRaymer=sqrt(2*W/rho/S*sqrt(AK/3/cD0));

     % represent Prequired as AA*U**3 + BB/U
% where
AA=0.5*rho*S*cD0;
BB=W^2/0.5/rho/S/pi/AR/Oswald;

     % opt Ustar to minimize Preq
Ustar=sqrt(BB/3/AA);
Ustar=sqrt(Ustar);

     % associated Mach number
AMstar=Ustar/spdsnd;

if AMstar > AMmax
   Uclimb=AMmax*spdsnd;
else
   Uclimb=Ustar;
end

     % Find cLstar  (based on opt speed)
     cLstar=W/0.5/rho/Uclimb^2/S;
     cL = cLstar;
%if cLstar > cLoptcLcD

%    cL=cLoptcLcD;

%end


     % Find corresponding speed

     Uclimb=sqrt(2*W/rho/S/cL);

cD=cD0+AK*cL^2;

     % find drag
     D = 0.5*rho*Uclimb^2*S*cD; %lb

% find power available
     P = 550*throttle*Np*Pmax*(rho/densSL)^AnPower; %ft-lb/sec

     if (P-D*Uclimb) <= 0
        P = D*Uclimb + 10;
     end

     % Find rate of climb dh/dt

     dhdt=(P-D*Uclimb)/W; %ft/sec


     % find the time it takes to climb an amount dh

     dt=dh/dhdt; %sec


%find climb angle gamma
```

```
        singamma=dhdt/Uclimb;
□

□
   % gamma in degrees
□
   gamma=asin(singamma)*180/pi;
□
   cosgamma=sqrt(1-singamma^2);
□

   %fuel burnt
        dWdt=-BSFC*P/(3600*550); %lb/sec
   dW=dWdt*dt; %lb

        % modify weight
        W=W+dW;

        % distance
   distance=distance+Uclimb*dt*cosgamma;

   % time
   time=time+dt;
end

Wout=W;
```

C.14 Cruise.m

This file performs the cruise leg analysis and is called directly from *Mission.m*.

```
% Cruise.m
□
% Performs a complete cruise analysis
□

fprintf(fid4,'\n');
fprintf(fid4,'\n Cruise:');
fprintf(fid4,'\n');

IN = MissIN(:,LegNumber+1);
□
option = IN(1);
□
altitude = IN(2);
□
Dist = IN(3)*5280; %miles to feet
BSFC = IN(4);
Np = IN(5);
Cruisespeed = IN(6);
throttle = IN(7);
□
RCcruise = IN(9);
□
cLfraction = IN(10);
□
```

```
cLratio = IN(11);
□
Atmos;
□

□
% Set Cruise Conditions for Smetana Stability Analysis
□
RHO = rho;
□
sound = spdsnd;
□

□
if option == 2
% Constant airspeed cruise condition

else
   % Use Breguet range equations, Raymer pg.105
   Wout = Win*exp(-Dist*(BSFC/(3600*550*Np))/LDmax);

   % Rate of Climb at Beginning of Cruise
   sigma = rho/densSL;
   Power = Pmax*sigma^AnPower; %hp
   AA = Np*550*Power/Win;
   BB = 0.5*rho*S*cD0/Win;
   CC = Win/pi/AR/Oswald*2/rho/S;
   Uopt = sqrt(sqrt(CC/3/BB));
   RC = AA-BB*Uopt^3-CC/Uopt;

   % Optimum Speed at Cruise
   cLstar = sqrt(pi*AR*Oswald*cD0);
   if cLstar > cLratio*cLmaxClean
      cLstar = cLratio*cLmaxClean;
   end
   cLstar = cLstar*cLfraction;
   Speed = sqrt(2*Win/S/cLstar/rho);
   MachCruise = Speed/spdsnd;
   cL = cLstar;
   cD = cD0+cL^2/pi/AR/Oswald;
   Drag = 0.5*rho*Speed^2*S*cD;

   % Power Required
   Preq = Drag*Speed; %ft-lb/sec
   % Find Max Speed
   Power = Power*550;
   Ustart = Speed;
□
   Uend = 20*Ustart;
□
   Maxspeed;
□

□
   % Power Available
□
   Pcruise = throttle*Power*Np; %ft-lb/sec
□
   Ucr = WindMax;
□
   Pwind = 0.5*rho*Ucr^3*S*CD0+Win^2/.5/rho/Ucr/S/pi/AR/Oswald; %ft-lb/sec

   % Rate of climb must be greater than desired rate of climb
```

```
    gCR(1,1) = 1-RC/RCcruise;
    % Power available must be greater than power required
    gCR(2,1) = Preq/Pcruise-1;
    % MachCruise must be less than UpperMach
    gCR(3,1) = MachCruise/UpperMach -1;
    % Max speed must be greater than max wind speed
    gCR(4,1) = WindMax/Umax-1;

end
U = Cruisespeed;
M = U/sound;
% Dynamic Pressure
qD = 0.5*RHO*U^2;

WratiosA(LegNumber,1)=Wout/Win;
Win = Wout;

gmission = [gmission;gCR];
fprintf(fid4,'\n Optimum Cruise Speed, ft/s        = %4.6f',Speed);
fprintf(fid4,'\n Mach Number                       = %4.6f',MachCruise);
fprintf(fid4,'\n Optimum Lift Coefficient          = %4.6f',cLstar);
fprintf(fid4,'\n Cruise Drag Coefficient           = %4.6f',cD);
fprintf(fid4,'\n Power Available, hp               = %4.6f',Pcruise/550);
fprintf(fid4,'\n Power Required, hp                = %4.6f',Preq/550);
fprintf(fid4,'\n Rate of Climb, ft/s               = %4.6f',RC);
fprintf(fid4,'\n Max Speed at Cruise Altitude,ft/s = %4.6f',Umax);
```

C.15 Loiter.m

This file performs the loiter leg analysis and is called directly by *Mission.m*.

```
% Loiter.m
%
% This file calculates the fuel burn during loiter using the Breguet endurance
% equation

fprintf(fid4,'\n');
fprintf(fid4,'\n Loiter:');
fprintf(fid4,'\n');

IN = MissIN(:,LegNumber+1);
%
option = IN(1);
%
altitude = IN(2);
%
Time = IN(3)*60; %Convert to seconds
%
BSFC = IN(4);
%
Np = IN(5);
%
Vel = IN(6);
throttle = IN(7);
RCloiter = IN(9);
Atmos;
```

```
% Implement Breguet endurance equation
Wout = Win*exp(-Time*(BSFC*Vel/(3600*550*Np))/(0.866*LDmax));

% Optimum Speed at Loiter
sigma = rho/densSL;
cLstar = sqrt(3*pi*AR*Oswald*cD0);
if cLstar > cLratio*cLmaxClean
   cLstar = cLratio*cLmaxClean;
end
cLstar = cLstar*cLfraction;
Speed = sqrt(2*Win/S/cLstar/rho);
MachLoiter = Speed/spdsnd;
cL = cLstar;
cD = cD0+cL^2/pi/AR/Oswald;
Drag = 0.5*rho*Speed^2*S*cD;

% Power Required
Preq = Drag*Speed; %ft-lb/sec
Power = 550*Pmax*sigma^AnPower; %ft-lb/sec
% Find Max Speed
Ustart = Speed;
Uend = 20*Ustart;
Maxspeed;

% Power Available
Ploiter = throttle*Power*Np; %ft-lb/sec
Ulo = WindMax;
Pwind = 0.5*rho*Ulo^3*S*CD0+Win^2/.5/rho/Ulo/S/pi/AR/Oswald; %ft-lb/sec

% Power available must be greater than power required
gLT(1,1) = Preq/Power-1;
% MachLoiter must be less than UpperMach
gLT(2,1) = MachLoiter/UpperMach-1;
% Max speed must be greater than max wind speed
gLT(3,1) = WindMax/Umax-1;

WratiosA(LegNumber,1)=Wout/Win;
Win = Wout;

gmission = [gmission;gLT];
fprintf(fid4,'\n Optimum Loiter Speed, ft/s        = %4.6f',Speed);
fprintf(fid4,'\n Mach Number                       = %4.6f',MachLoiter);
fprintf(fid4,'\n Optimum Lift Coefficient          = %4.6f',cLstar);
fprintf(fid4,'\n Loiter Drag Coefficient           = %4.6f',cD);
fprintf(fid4,'\n Power Available, hp               = %4.6f',Ploiter/550);
fprintf(fid4,'\n Power Required, hp                = %4.6f',Preq/550);
fprintf(fid4,'\n Max Speed at Loiter Altitude,ft/s = %4.6f',Umax);
```

C.16 Landing.m

This file performs the complete landing analysis and is called by *Mission.m*.

```
% Landing.m
□
% This file allows for two options for the calculation of fuel burn during
□
% a combined descent and landing.
```

```
fprintf(fid4,'\n');
fprintf(fid4,'\n Descent and Landing:');
fprintf(fid4,'\n');

IN = MissIN(:,LegNumber+1);
□
option = IN(1);
□
altitude = IN(2);
□
SpeedLand = IN(6);
□
RunwayLand = IN(8);
□
gammabar = IN(12);
□
deltaN = IN(13);
□
groundFrcition = IN(14);
Hland = IN(15);

cLTD = cLmax/sqrt(1.21);
Atmos;

if option == 2

else
   % Statistical landing analysis
   % Descend and land (see Raymer, Table 3.2)   W5 --> W6
   Wout=.995*Win;

   speed = sqrt(2*Win/S/rho/cLTD);
   % Average landing speed
   Avspeed = speed/sqrt(2);
   cLrunway = 0.8*cLTD;
   Lift = .5*rho*Avspeed^2*S*cLrunway;
   cD = (1+cDFlapRatio+cDLGratio)*cD0+cL^2/pi/AR/Oswald*groundeffect;
   Drag = 0.5*rho*Avspeed^2*S*cD;
   Friction = (Win - Lift)*groundFriction;
   % Average decceleration upon landing
   avDecel = (Drag + Friction)/Win*GRAVIT;
   abaroverg = avDecel/GRAVIT;
   % Calculate needed runway length
   Sland = 1/abaroverg*(1-gammabar^2/deltaN)+gammabar/deltaN;
   Sland = Sland*1.69*Win/S/Hland/rho/GRAVIT/cLmax;
   Sland = Sland+1/gammabar;
   Sland = Sland*Hland;
   % Add FAA Margin of Safety for pilot technique
   Sland = Sland*1.67;

   % Required runway for landing must be less than actual runway
   gLD(1,1) = Sland/RunwayLand-1;
   % Landing speed must be less than or equal to desired landing speed
   gLD(2,1) = Avspeed/SpeedLand-1;

end
WratiosA(LegNumber,1)=Wout/Win;
Win = Wout;

gmission = [gmission; gLD];
fprintf(fid4,'\n Average Landing Speed, ft/s       = %4.6f',Avspeed);
fprintf(fid4,'\n Runway Lift Coefficient           = %4.6f',cLrunway);
```

```
fprintf(fid4,'\n Runway Drag Coefficient          = %4.6f',cD);
fprintf(fid4,'\n Needed Runway Length, ft         = %4.6f',Sland);
```

C.17 Atmos.m

This file determines the atmospheric properties for a given altitude. It is called throughout the mission analysis by each of the mission leg analysis files.

```
%Atmos.m
□
%determines atmospheric properties up to 25 km
□

h = altitude*.3048;
□

□
% MKS units
□

□
rho0 =1.225;
t0 = 288.16;
p0 = 1.01325*10^5;
visc0 = 1.7894*10^-5;

if h > 25000
   fprintf(fid1,' Altitude greater than 25000 meters. stop. h=',h);
   error('Altitude greater than 25000 meters');
end

if h <= 11000
   t = t0 - (6.5*10^-3)*h;
   rho = rho0*(1-h/44300)^4.256;
else
   t = t0-71.5;
   rho11000 = rho0*(1-11000/44300)^4.256;
   rho = rho11000*exp((11000-h)/6340);
end
visc = visc0*(t/t0)^0.75;
spdsnd = 20.066*sqrt(t);
delta = rho/rho0;

% Convert to English units!
□
rho = rho*0.00194032653;
visc = visc*.020894065;
spdsnd = spdsnd/.3048;
t = 1.8*t; % deg Rankine
```

C. 18 Maxspeed.m

This file determines the maximum speed of the aircraft at the specified conditions. It is called during the cruise and loiter mission leg analysis files.

```
% Maxspeed.m
% Determines maximum airspeed
□

□
% find max speed
□
n=100;
□
Umax=0;
□
dU=(Uend-Ustart)/n;
□
Uo=Ustart-dU;
□

□
for i=1:1:n+1
□
      Uo=Uo+dU;
□
       Pavailable=Np*Power*throttle;
□
        CLspeed = Win/(.5*S*rho*Uo^2);
        Prequired=(0.5*rho*S*Uo^2*(cD0 + CLspeed^2/(AR*Oswald*pi)))*Uo;
        Excess=Pavailable-Required;

   if i==1
      ExcessOld=Excess;
      Uold=Uo;
        else
              ExcessNew=Excess;

      if (ExcessOld >= 0)&(Excess<0)
        % interpolate
                    Umax=Uold+(Uo-Uold)*(0-ExcessOld)/(ExcessNew-ExcessOld);
      else
              ExcessOld=Excess;
        Uold=Uo;
              end
        end
end

if Umax == 0
   Umax=999;
end
```

C.19 Weights.m

This file performs the component weight breakdown analysis and determines the moments of inertia. It is called directly by *analyze.m*.

```
% Weights.m
%
% This program acts as a subroutine to the main design analysis code.  It serves to
%
% estimate the aircraft component weights, total aircraft weight, and the location of
%
% the center of gravity for an initial configuration.  Equations and estimates are
% taken directly from Raymer's text, "Aircraft Design: A Conceptual
% Approach."
%*****************************************************************************

% Calculate component weights in pounds and masses in slugs
% Wing
Wwing = 0.036*S^0.758*Wfw^0.0035*(AR/cos(SA)^2)^0.6*qD^0.006*TR^0.04*...
   (100*tc/cos(SA))^-0.3*(Nz*Wdg)^0.49;
Mwing = Wwing/GRAVIT;

% Horizontal Tail
Wht = 0.016*(Nz*Wdg)^0.414*qD^0.168*ST^0.896*(100*tc/cos(SA))^-0.12*...
   (AR/cos(SAH)^2)^0.043*TRT^-0.02;
Mht = Wht/GRAVIT;

% Vertical Tail
Wvt = 0.073*(1+0.2*HtHv)*(Nz*Wdg)^0.376*qD^0.122*SV^0.873*(100*tc/cos(SAvt))^-0.49*...
   (AR/cos(SAvt)^2)^0.357*TRvt^0.039;
Mvt = Wvt/GRAVIT;

% Fuselage
Wpress = 11.9 + (Vpr*Pdelta)^0.271; %Check units?  inches and feet problem.
Wfuse = 0.052*Sf^1.086*(Nz*Wdg)^0.177*LT1^-0.051*(LB/WFUS)^-0.072*qD^0.241+Wpress;
Mfuse = Wfuse/GRAVIT;

% Main Landing Gear
Wmain = 0.095*(Nl*Wl)^0.768*(Lm/12)^0.409;
Mmain = Wmain/GRAVIT;

% Nose Landing Gear
Wnose = 0.125*(Nl*Wl)^0.566*(Ln/12)^0.845;
Mnose = Wnose/GRAVIT;

% Total Installed Engine Weight
Wengine = 2.575*Wen^0.922*Nen;
Mengine = Wengine/GRAVIT;

% Fuel System
Wfuel = 2.49*(Vt/Nt)^0.726*(1/(1+Vi/Vt))^0.363*Nt^0.242*Nen^0.157;
Mfuel = Wfuel/GRAVIT;

% Flight Controls
Wcontrol = 0.053*LB^1.536*B^0.371*(Nz*Wdg*10^-4)^0.8;
Mcontrol = Wcontrol/GRAVIT;

% Hydraulics
Whydra = 0.001*Wdg;
```

```
Mhydra = Whydra/GRAVIT;

% Avionics
Wavion = 2.117*Wuav^0.933;
Mavion = Wavion/GRAVIT;

% Electrical
Welec = 12.57*(Wfuel+ Wavion)^0.51;
Melec = Welec/GRAVIT;

% Air Conditioning and Anti-ice
Wac = 0.265*Wdg^0.52*Np^0.68*Wavion^0.17*M^0.08;
Mac = Wac/GRAVIT;

% Furnishings
Wfurn = 0.0582*Wdg-65;
Mfurn = Wfurn/GRAVIT;

% Calculated Weight Using Raymer's Statistical Group Weights Method
Wtotal = Wwing+Wht+Wvt+Wfuse+Wmain+Wnose+Wengine+Wfuel+Wcontrol+Whydra+...
   Wavion+Welec+Wac+Wfurn+Wppass*Npassengers+Wpcrew*Ncrew+Wpayload;
Mtotal = Wtotal/GRAVIT;
MS =Mtotal;

% Considered adding capability for cg travel
% Calculated Empty Weight
% Wempty = Wtotal - Wfuel;

% Calculate Aircraft C.G.
Xcg = (CGwing*Wwing+CGht*Wht+CGvt*Wvt+CGfuse*Wfuse+CGmgr*Wmain+CGngr*Wnose+...
   CGeng*Wengine+CGfuel*Wfuel+CGctrl*Wcontrol+CGhydr*Whydra+CGelec*Welec+...
   CGavion*Wavion+CGac*Wac+CGfurn*Wfurn)/(Wwing+Wht+Wvt+Wfuse+Wmain+Wnose+...
   Wengine+Wfuel+Wcontrol+Whydra+Welec+Wavion+Wac+Wfurn);

% Calculate Moments of Inertia
% Values from Table 16.1, pg.444 in Raymer...dependent on aircraft type
% Values modified to match known moments of inertia
Rx = 0.277826;
Ry = 0.46501;
Rz = 0.46754;
% Equations from Raymer pg.443
IXX = B^2*Wtotal*Rx^2/(8*GRAVIT);
IYY = LB^2*Wtotal*Ry^2/(8*GRAVIT);
IZZ = ((B+LB)/2)^2*Wtotal*Rz^2/(8*GRAVIT);
IXZ = 0; % By assumption
```

C.20 LandingGear.m

This file determines the tip-over and ground clearance angles. It is called directly by *analyze.m*.

```
% LandingGear.m
% This routine was developed to calculate the tip over angle, longitudinal ground
% clearance criterion, and the landing gear static loads for constraint purposes.
% Developed 5 June 2000
```

```
% Assume length of main landing gear includes tire radius (ft)
% Calculate tip over angle (ref. Figure 9.1a Roskam)
Zgear = 1/2*Hgear+ZT+Lm/12;
Xgear = CGmgr-XM;
% This angle is typically 15 deg...we will look for tipangle between 14 and 16 deg.
% Two design options: 1. Decrease main gear length
%                                                    2. Move main gear aft
tipangle = atan(Xgear/Zgear)*180/(pi);

% Longitudinal ground clearance: Approximately 15 degrees for general aviation a/c
% Two design options: 1. Increase main gear length
%                                                    2. Move main gear aft
clearance = atan(Zgear/(LB-CGmgr))*180/(pi);

% We want the landing gear to take 10% of the TOGW on the nose gear and 90% on
% the main gear (reg. Figure 9.2a Roskam)
Pnose = Wguess*Xgear/(CGmgr-CGngr); %Equation 9.1 Roskam
Pmain = Wguess*(XM-CGngr)/((CGmgr-CGngr)); %Equation 9.2 Roskam
```

C.21 VdocLongStab.m

This file is slightly modified from the original Smetana code. It calculates the lontiudinal stability derivatives. It is called by *analyze.m*.

```
%************************************************************************
%  April 19, 2000
%------------------------------------------------------------------------
%  VdocLongStab.m     Longitudinal Stability Derivatives
%------------------------------------------------------------------------
% General Aviation, Low Subsonic Aircraft
%
% For the Longitudinal Case this program calculates:
%       1) Non-Dimensional Stability Derivatives
%       2) Dimensional Stability Derivatives
%
% Written by Prof. Frederick Smetana
%
% Derivation of Equations of Motion is based on
% 'Dynamics of the Airframe', Bureau of Aeronautics Report, AE-61-4II.
%
%        ASSUMPTIONS:
%
%       1) THE AIRFRAME IS ASSUMED TO BE A RIGID BODY.
%       2) THE EARTH IS ASSUMED TO BE FIXED IN SPACE, AND, UNLESS
%          SPECIFICALLY STATED OTHERWISE, THE EARTH'S ATMOSPHERE IS
%          ASSUMED TO BE FIXED WITH RESPECT TO THE EARTH.
%       3) THE MASS OF THE AIRPLANE IS ASSUMED TO REMAIN CONSTANT FOR
%          THE DURATION OF ANY PARTICULAR DYNAMIC ANALYSIS.
%       4) THE X-Z PLANE IS ASSUMED TO BE A PLANE OF SYMMETRY.
%       5) THE DISTURBANCES FROM THE STEADY FLIGHT CONDITION ARE ASSUMED
%          TO BE SMALL ENOUGH SO THAT THE PRODUCTS AND SQUARES OF THE
%          CHANGES IN VELOCITIES ARE NEGLIGIBLE IN COMPARISON WITH THE
%          CHANGES THEMSELVES.  ALSO, THE DISTURBANCE ANGLES ARE ASSUMED
%          TO BE SMALL ENOUGH SO THAT THE SINES OF THESE ANGLES MAY BE
%          SET EQUAL TO THE ANGLES AND THE COSINES SET EQUAL TO ONE.
%          PRODUCTS OF THESE ANGLES ARE ALSO APPROXIMATELY ZERO AND CAN
```

```
%          BE NEGLECTED.  AND, SINCE THE DISTURBANCES ARE SMALL, THE
%          CHANGE IN AIR DENSITY ENCOUNTERED BY THE AIRPLANE DURING ANY
%          DISTURBANCE CAN BE CONSIDERED TO BE ZERO.
%       6) DURING THE STEADY FLIGHT CONDITION, THE AIRPLANE IS ASSUMED
%          TO BE FLYING WITH WINGS LEVEL AND ALL COMPONENTS OF VELOCITY
%          ZERO EXCEPT U SUB 0.  W SUB 0 = 0 BECAUSE THE STABILITY AXES
%          WERE CHOSEN AS THE REFERENCE AXES.
%       7) THE FLOW IS ASSUMED TO BE QUASI-STEADY.
%
% The Pertinent Aircraft Characteristics are Defined as Follows:
%
% Flight Conditions (Ref Flight)
%
%     RHO - Density at altitude (slug/ft^3)
%     U - Aircraft speed (ft/s)
%     GAMMA - Flight Path Angle (Deg)
%     GRAVIT - Acceleration of Gravity
%     GCOSGM and GSINGM are the Products of the Acceleration Due to
%     GRAVITY (Assumed = 32.2 ft/s^2 for this Altitude Range) and the
%     COSINE and SINE Respectively of the Initial Flight Path Angle,
%     GAMMA, (Usually Zero for Level Flight).
%     MS - Mass of the Aircraft (slugs)
%     IYY - Moment of Inertia About the Y-Axis (Slug-ft^2)
%     THRUST - Aircraft Thrust (lb)
%     ZJ - Perpendicular Distance from the Center of Gravity to the Thrust
%              Line (Positive for the C.G. Above the Thrust Line, ft)
%     COSXZ - Cosine of the Angle Made Between the Thrust Axis and
%             the Wind Axis
%     SINXZ - Sine of the Angle Made Between the Thrust Axis and
%             the Wind Axis
%     S - Wing Area (ft^2)
%     ST - Horizontal Tail Area (ft^2)
%     B - Wing Span (ft)
%     BT - Horizontal Tail Span (ft)
%     EFF - Efficiency of the Horizontal Tail
%     CHE - Root Chord of the Elevator (ft)
%     TR - Taper Ratio of the Wing
%     TRT - Taper Ratio of the Horizontal Tail
%     ITAIL - Incidence of the Horizontal Tail (Deg)
%     CLA2DW - 2-D Lift Curve Slope of the Wing (1/rad)
%     CLA2DT - 2-D Lift Curve Slope of the Horizontal Tail (1/rad)
%     CD0 - Parasite Drag Coefficient
%     CDA2DW - 2-D Drag Curve Slope of the Wing (1/rad)
%     ALPHA - Wing Angle og Attack (Deg)
%     IWING - Incidence Angle of the Wing (Deg)
%     LT - Distance from the C.G. to the Quarter-Chord of the Horizontal
%          Tail (ft)
%     LT1 - Distance from the Wing Quarter-Chord to the Quarter-Chord of
%           the Horizontal Tail (ft)
%     XA - Chordwise Distance from the C.G. to the Wing Aerodynamic
%          Center (Positive for C.G. Behinde the Wing A.C., ft)
%     ZA - Vertical Distance Ffrom the C.G. to the Wing A.C.
%                        (Positive for the C.G. Below the Wing A.C., ft)
%     LB - Length of the Fuselage (ft)
%     WFUS - Maximum Width of the Fuselage (ft)
%     XFUS - Distance from the Nose to the Wing Quarter-Chord (ft)
%     XCG - Chordwise Distance from the C.G. to the Wing Quarter-Chord
%           (Positive for the C.G. Ahead of the Wing Quarter-Chord, ft)
%     ZT - Vertical Distance from the C.G. to the Thrust Line
%          (Positive for Thrust Line Below the C.G., ft)
%     SELEV - Area of the Elevator (ft^2)
%     AR - Wing Aspect Ratio
%     ART - Horizontal Tail Aspect Ratio
```

```
%     CH -  MAC of the Wing
%     CHT - Horizontal Tail Chord
%     TAU - Correction Factor for Induced Angle of the Wing
%     TAU1 - Correction Factor for Induced Angle of the Wing for TR = 1
%     E1 - Induced-Angle Span Efficiency Factor of the wing
%     CLAW - 3-D Wing Lift Curve Slope (1/rad)
%     TAUT - Correction Factor for Induce Angle of the Horizontal Tail
%     TAU1T - Correction Factor for Induced Angle of the Horizontal Tail
%               if TRT = 1
%     E1T - Induced-Angle Span Efficiency Factor of the Horizontal Tail
%     DELTA - Correction Factor for Induced Drag of the Wing
%     DELTA1 - Correction Factor for Induced Drag of the Wing for TR = 1
%     E - Oswald's Span Efficiency Factor of the Wing
%     CLW - 3-D Wing Lift Coefficient
%     DW - Downwash Angle at the Horizontal Tail
%     ALPHAT - Angle of Attack at the Horizontal Tail
%     SRATIO - Ratio of the Elevator Area to Horizontal Tail Area
%     DADDE - Elevator Efficiency Factor; Change in the Tail Angle
%             of Attack Due to Elevator Deflection
%     CLAT - 3-D Horizontal Tail Lift-Curve Slope (1/rad)
%     CLT - 3-D Horizontal Tail Lift Coefficient Needed for Equilibrium
%     ALPHAR - Angle of Attack Required to Acheive the Necessary
%              Tail Lift Coefficient
%     DALPHA - Effective Angle of Attack Which Must Be Supplied by
%              the Elevator to Obtain Equilibrium
%     DELTAE - Elevator Deflection Required for Equilibrium
%     CL - Total 3-D Airplane Lift Coefficient
%     CD - Total 3-D Airplane Drag Coefficient
%     CM - Total 3-D Airplane Pitching Moment
%     CT - Total 3-D Airplane Thrust Coefficient
%     DEDA - Change in Downwash With Wing Angle of Attack
%     CLA - Total 3-D Airplane Lift Curve Slope (1/rad)
%     CDA - 3-D Airplane Drag Curve Slope (1/rad)
%     CMAW - Change in Wing Pitching Moment Due to Angle of Attack (1/rad)
%     CMAT - Change in Horizontal Tail Pitching Moment Due to Angle
%            of Attack (1/rad)
%     XLBo - Ratio of the Distance From the Nose to the Wing Quarter-Chord
%                              to the Total Fuselage Length
%     KF - Empirical Factor for Fuselage or Nacelle Contributions to CMA
%     CMAFUS - Change in Fuselage Pitching Moment Due to Angle of Attack (1/rad)
%     CMA - Total Airplane Change in Pitching Moment Due to Angle of
%           Attacl (1/rad)
%     CLDA - Change in Airplane Lift Coefficient With Rate of Change
%            of Angle of Attack
%     CDDA - Change in Airplane Drag Coefficient With Rate of Change of
%            Angle of Attack
%     CMDA - Change in Airplane Pitching Moment Coefficient With Rate of
%            Change of Angle of Attack
%     CLQ - Change in Airplane Lift Coefficient With Pitching Rate
%     CDQ - Change in Airplane Drag Coefficeint With Pitching Rate
%     CMQ - Change in Airplane Pitching Moment Coefficient With Pitching Rate
%     CLDE2D - 2-D Elevator Effectiveness
%     CLDE - Change in Airplane Lift Coefficient with Elevator Deflection (1/rad)
%     CDDE - Change in Airplane Drag Coefficient with Elevator Deflection (1/rad)
%     CMDE - Change in Airplane Pitching Moment Coefficient with Elevator
%            Deflection (1/rad)
%     CLU, CDU, CMU, CTU - Change in Lift, Drag, Pitching Moment, and
%                          Thrust Coefficients with Perturbation Airspeed
%     CTRPM - Change in Thrust Coefficient with Engine Speed
%
%     ALTHOUGH THE PROGRAM DOES NOT CALCULATE THE STABILITY DERIVATIVES
%     DUE TO FLAP DEFLECTION THESE DERIVATIVES CAN BE CALCULATED AND
%     THEN READ INTO THE PROGRAM AS CLDE, CDDE, AND CMDE.
```

```
% Open output file
fid1 = fopen('LongStabOut.txt','w');

fprintf (fid1, 'Solution for Smetana via MATLAB');
fprintf (fid1, '\n');

% Echo Check of Wing Input Data
fprintf(fid1,'\n Flight Condition') ;
fprintf(fid1,'\n Density, slug/ft^3      = %4.6f',RHO);
fprintf(fid1,'\n Velocity, ft/s          = %4.2f',U);
fprintf(fid1,'\n Flight Path Angle, rad  = %4.2f',GAMMA);
fprintf(fid1,'\n Gravity, ft/s^2         = %4.2f',GRAVIT);
fprintf(fid1,'\n Mass, slugs             = %4.2f',MS);
fprintf(fid1,'\n Iyy, slug*ft^2          = %4.2f',IYY);
fprintf(fid1,'\n');
fprintf(fid1,'\n Wing Data');
fprintf(fid1,'\n Wing Area, ft^2         = %4.2f',S);
fprintf(fid1,'\n Wing Span, ft           = %4.2f',B);
fprintf(fid1,'\n Taper Ratio             = %4.2f',TR);
fprintf(fid1,'\n Wing incidence, deg     = %4.2f',IWING);
fprintf(fid1,'\n a.c. x-coord., ft       = %4.2f',XA);
fprintf(fid1,'\n a.c. z-coord., ft       = %4.2f',ZA);
fprintf(fid1,'\n cL-alpha Wing 2D        = %4.2f',CLA2DW);
fprintf(fid1,'\n cD-alpha Wing 2D        = %4.2f',CDA2DW);
fprintf(fid1,'\n') ;
fprintf(fid1,'\n Horizontal Tail Data');
fprintf(fid1,'\n Tail Area, ft^2         = %4.2f',ST);
fprintf(fid1,'\n Tail Span, ft           = %4.2f',BT);
fprintf(fid1,'\n Taper Ratio             = %4.2f',TRT);
fprintf(fid1,'\n Tail incidence, deg     = %4.2f',ITAIL);
fprintf(fid1,'\n Lt cg to .25 ctail      = %4.2f',LT);
fprintf(fid1,'\n Lt .25 wing to .25 tail = %4.2f',LT1);
fprintf(fid1,'\n zT, ft                  = %4.2f',ZT);
fprintf(fid1,'\n Tail efficiency         = %4.2f',EFF);
fprintf(fid1,'\n cL-alpha Tail 2D        = %4.2f',CLA2DT);
fprintf(fid1,'\n');
fprintf(fid1,'\n Elevator Data');
fprintf(fid1,'\n Elevator Area, ft^2     = %4.2f ',SELEV);
fprintf(fid1,'\n Elevator Chord, ft      = %4.2f ',CHE) ;
fprintf(fid1,'\n Elevator Spna, ft       = %4.2f ',BE);
fprintf(fid1,'\n') ;
fprintf(fid1,'\n Fuselage Data');
fprintf(fid1,'\n Body length, ft         = %4.2f ',LB);
fprintf(fid1,'\n Max diameter, ft        = %4.2f ',WFUS) ;
fprintf(fid1,'\n');
fprintf(fid1,'\n Configuration Data');
fprintf(fid1,'\n Location of a.c., ft    = %4.2f ',XFUS);
fprintf(fid1,'\n Location of c.g., ft    = %4.2f ',XCG) ;
fprintf(fid1,'\n Zero-lift drag coeff    = %4.2f ',CD0) ;
fprintf(fid1,'\n') ;
fprintf(fid1,'\n Thrust Data');
fprintf(fid1,'\n Thrust line from c.g    = %4.2f ',ZJ);
fprintf(fid1,'\n COSXz                   = %4.2f ',COSXZ);
fprintf(fid1,'\n SINXz                   = %4.2f ',SINXZ);
K=1;
% The original Smetana Code used mean geometric chords and
% not mean aerodynamic chords. The original equations are commented
% out and replaced by eqs. for mean aerodynamic chords.

% note: ch here is a chord and NOT a hinge moment coefficient.
```

```
% original equations (canceled)
%     CH=S/B
%     CHT=ST/BT

% mean aerodynamic chord of wing

% Sw = (cr + ct) /2 *b, with tip chord ct=TR*cr

% Find root chord and tip chord
CWROOT = (2*S)/B/(1+TR);
CWTIP = CWROOT*TR;

% Use Schmidt p. 9, eq. 1.13
%CH = (2/3)*(CWROOT + CWTIP - (CWROOT*CWTIP/(CWROOT+CWTIP)));
CH = MACwing;
% Mean aerodynamic chord of horizontal tail

% St = (cr tail+ ct tail) /2 *bt, with tip chord ct=TRT*cr

% Find root chord and tip chord
CTROOT = (2*ST)/BT/(1+TRT);
CTTIP = CTROOT*TRT;

% Use Schmidt p. 9, eq. 1.13
%CHT = (2/3)*(CTROOT + CTTIP - (CTROOT*CTTIP/(CTROOT+CTTIP)));
CHT=MACht;

fprintf(fid1,'\n');
fprintf(fid1,'\n Wing MAC, ft            = %4.2f ',CH);
fprintf(fid1,'\n Tail MAC, ft            = %4.2f ',CHT);

% Dynamic Pressure
qD = 0.5*RHO*U^2;

fprintf(fid1,'\n');
fprintf(fid1,'\n ****Internal Calculations. Interim Results**** ');
fprintf(fid1,'\n Dynamic Pressure, lb/ft^2        = %4.6f ',qD);

% Check values of taper ratio for wing
% For most accurate results TR should be between 0 and 1

if (TR < 0 | TR > 1)
   disp('TR = ');
   disp(TR);
   disp('ERROR: TR IS OUTSIDE DESIRABLE RANGE OF 0.0 TO 1.0');
   disp('WHEN CALCULATING TAU OR DELTA');
end

% Make sure that TR and AR are within values for which aerodynamic
% effects are given. See Smetana page 59 Fig. 12.
% Note that the Aspect ratio range for that figure (for TR=1) is
% between 3 and 12.

if (((AR < 3) & (TR == 1))|((AR > 12) & (TR == 1)))
   disp('AR = ');
   disp(AR);
   disp('ERROR: AR IS OUTSIDE DESIRABLE RANGE OF 3.0 TO 12.0');
   disp('FOR TR = 1.0 WHEN CALCULATING TAU1 OR DELTA1');
end


if TR == 1
   % data for TR=1 (Fig. 12, p. 59 in Smetana) AR between 3 and 12.
```

```
        TAU1 = 0.0000297115*AR^4 - 0.000811747*AR^3 + 0.0071717*AR^2 -...
        0.00298853*AR + 1.07739;
        E1 = 1/TAU1;

else    %if (AR <= 7 & AR >= 5)
        %  The case where TR is not 1. Fig. 11 page 59 of Smetana
        % in this case AR must be around 6

        % make sure AR is around 6
   TAU = 6526.0*TR^13-33936.5*TR^12+65708.3*TR^11-40275.9*TR^10-...
      55022.8*TR^9+134075.0*TR^8-128452.0*TR^7+71152.4*TR^6-24169.6*TR^5+...
      4910.63*TR^4-541.19*TR^3+27.1966*TR^2-1.31155*TR+0.170212;
   E1 = 1/(1 + TAU);
   %     else
   % Modification made 29 Mar 00.  The above restriction on AR does not appear
   % in the Smetana code.  Was added by Prof. Livne for AA516.

%     disp('AR = ');
%       disp(AR);
%       disp('ERROR: AR IS OUTSIDE DESIRABLE RANGE OF 5.0 TO 7.0');
%       disp('FOR TR ~= 1.0');
%error('Program stopped');

end

% Find CLalpha-wing page 58 eq. at center of page

% notes: The data for correction of 2D cLalpha to 3D cLalpha is incomplete.
% Only the cases TR=1 for AR from 3 to 12
% OR
% AR=6.28 for TR from 0 to 1
% are considered.
% Thus this can be used for airplanes of TR=1
% or
% airplane with AR of about 6 and varying TR.
% for more combinations of TR and AR see Perkins and Hage page 84-85
% equation 2-66 and fig. 2-55.

% pay attention to units. The lift curve slopes are input as 1/deg
% the 3D result is in 1/rad.

CLAW = (57.3*CLA2DW)/(1 + CLA2DW*57.3/(pi*E1*AR));
fprintf(fid1,'\n Wing 3-D CL-alpha, 1/rad         = %4.6f ',CLAW);


% Tail 3D lift curve slope cLalpha-tail

% The tail is treated as a 3D wing

% Check that TRtail is between 0 and 1

if (TRT < 0 | TRT > 1)
   disp('TRT = ');
   disp(TRT);
   disp('ERROR: TRT IS OUTSIDE DESIRABLE RANGE OF 0.0 TO 1.0');
   disp('WHEN CALCULATING TAUT');
   %error('Program stopped');
end

% CHECK THAT IN THE CASE OF TR=1, AR IS Between 3 AND 12.

if ((ART < 3 & TRT == 1)|(ART > 12 & TRT == 1))
   disp('ART = ');
```

```
   disp(ART);
   disp('ERROR: ART IS OUTSIDE DESIRABLE RANGE OF 3.0 TO 12.0');
   disp('FOR TR = 1.0 WHEN CALCULATING TAU1T');
   %error('Program stopped');
end

% for tail 3D CLalpha-tail see comments regarding cLalpha of the wing.

% however, because the tail is influenced by the wing, a correction
% due to depsilon/dalpha must be added. That is needed not for the
% cLalpha of the tail, but for the contribution of the tail to the
% cLalpha of the airplane.

% See Smetana last Eq. on p. 63.

if (TRT == 1)

   % the case TRtail=1 and ARtail between 3 and 12.

   TAU1T = 0.0000297115*ART^4-0.000811747*ART^3+0.0071717*ART^2-0.00298853*ART...
        +1.07739;
        E1T = 1/TAU1T;

   % the case AR=6 and variable TR
else  %if (AR <= 7 & AR >= 5)
        %  The case where TR is not 1. Fig. 11 page 59 of Smetana
        % in this case AR must be around 6
        % make sure AR is around 6

   TAUT = 6526.0*TRT^13-33936.5*TRT^12+65708.3*TRT^11-40275.9*TRT^10-...
        55022.8*TRT^9+134075.0*TRT^8-128452.0*TRT^7+71152.4*TRT^6-...
        24169.6*TRT^5+4910.63*TRT^4-541.19*TRT^3+27.1966*TRT^2-1.31155*TRT+0.170212;
   E1T = 1/(1 + TAUT);
%     else
   % Modification made 29 Mar 00.  The above restriction on AR does not appear
   % in the Smetana code.  Was added by Prof. Livne for AA516.

%     disp('AR = ');
%       disp(AR);
%       disp('ERROR: AR IS OUTSIDE DESIRABLE RANGE OF 5.0 TO 7.0');
%       disp('FOR TR ~= 1.0');
%error('Program stopped');

end

%  for the wing find delta (see perkins & Hage p. 73 eq. 2-61, and fig.2-44

if TR == 1

   % the case of variable AR (3-12) at TR=1
   DELTA1 = -0.0000143113*AR^3+0.0000158924*AR^2+0.0113969*AR+.988661;
        E = 1/DELTA1;

else

        % the case of variable TR 90-1) at AR of about 6
        DELTA = 2.48637*TR^6-9.29906*TR^5+14.0692*TR^4-11.1199*TR^3+5.01493*TR^2-...
        1.24262*TR+0.141122;
        E = 1/(1+DELTA);

end

% Find CL
```

```
% -------

      % CL of airplane from Lift=Weight
      Weight = MS*GRAVIT;
  CL = Weight/qD/S;

      fprintf(fid1,'\n CL Airplane                          = %4.6f ',CL);

      % estimate tail lift coefficient: using moment balance about cg
      % (total pitching moment zero) and neglecting fuselage contribution,
      % the moment due to tail lift has to balance the moment due to wing lift.
      % Thus (Smetana page 40 2nd equation):

      %     CLT=CLW*XA*S/(LT*ST*EFF)

      % solve for CLW using LW+LT=Total Lift

      AUX = 1 + (XA*S/(LT*ST*EFF));
      CLW = CL/AUX;

      fprintf(fid1,'\n CL Wing                              = %4.6f ',CLW);

      % now find CLT

      CLT = CLW*XA*S/(LT*ST*EFF);

      fprintf(fid1,'\n CL Tail                              = %4.6f ',CLT);

      % now calculate depsilon/dalpha (Smetana, p. 69, last equation)

      % recall the wing cL is related to the strength of the wing vortex,
      % which, in turn, induces down wash over the tail

      DW = 20*CLW*(((1/TR)^0.3)/(AR^0.725))*(3.0*CH/LT1)^0.25;

      fprintf(fid1,'\n Wing Downwash on Tail (deg)          = %4.2f ',DW);

      % find clalpha of the wing and of the tail
      % Smetana p. 58 2nd eq.
      % cLalpha of wing
      CLAW = (57.3*CLA2DW)/(1 + CLA2DW*57.3/(pi*E1*AR));
      % cLalpha of tail
      CLAT = (57.3*CLA2DT)/(1 + CLA2DT*57.3/(pi*E1T*ART));

      % find angle of attack of the wing

      % using CLW=CLAW*ALPHAW (since CLAW is in 1/rad, we need
      % to convert ALPHAW to degrees

      ALPHAW = (CLW/CLAW)*180/pi;

      % set the airplane angle of attack to equal the wing AOA

      ALPHA = ALPHAW;

      fprintf(fid1,'\n Wing AOA, deg                        = %4.2f ',ALPHA);

      % tail angle of attack (Smetana p. 53, 2nd eq.)
      % (note that itail, iwing are real*8 variables)

      ALPHAT = ALPHA - IWING + ITAIL - DW;

      fprintf(fid1,'\n Tail AOA, deg                        = %4.2f ',ALPHAT);
```

```
        % ratio of elevator area to tail area
        % (Smetana p. 90, Fig. 20)

        SRATIO = SELEV/ST;

        % make sure that this ratio is between 0 and 0.7

        if (SRATIO < 0 | SRATIO > 0.7)
                disp('SRATIO = ');
                disp(SRATIO);
                disp('ERROR: SRATIO IS OUTSIDE DESIRABLE RANGE OF 0.0 TO 0.7');
                disp('WHEN CALCULATING DADDE');
                %error('Program stopped');
        end

        % dalphatail/ddeltaE
        DADDE = 21.7949*SRATIO^5-46.4744*SRATIO^4+36.9347*SRATIO^3-14.259*SRATIO^2+...
                3.70551*SRATIO-0.000057815;

        % for SRATIO between 0.7 and 1

        if (SRATIO > 0.7)
                DADDE = SRATIO;
        end

        if (SRATIO > 1)
                disp('ERROR: SRATIO > 1.0');
                %error('Program stopped');
        end

        % effective angle of attack of the tail (in degrees)

        ALPHAR = CLT/(CLAT/57.3);

        % difference between effective tail AOA and actual tail AOA

        DALPHA = ALPHAR-ALPHAT;

        %  Elevetor angle, deltaE, needed for balance

        DELTAE = DALPHA/DADDE;

        fprintf(fid1,'\n Elevator Angle deltaE, deg       = %4.6f ',DELTAE);


% Find Drag Coefficient CD
% ------------------------


        % drag polar equation with cD0=cDpi

        CD = CD0 + CL^2/(pi*E*AR);

        fprintf(fid1,'\n Oswald Coefficient (wing)        = %4.6f ',E);


fprintf(fid1,'\n Drag coefficient, cD          = %4.6f ',CD);

% Find Thrust force
% -----------------
```

```
% Thrust=Drag for straight and level flight at constant speed

THRUST = qD*S*CD;

% Pitching moment coefficient CM
%------------------------------

        % CM=0 in equilibrium flight, unless the thrust line is a distance
        % ZJ from the cg. In this case the aerodynamic moment CM has to balance
        % the moment due to thrust about the cg.

        % note: the thrust must actually be equal to the drag in constant speed
        % flight at constant altitude.

        CM = (THRUST/(0.5*RHO*U^2*S))*(ZJ/CH);

% Thrust Coefficient
% ------------------


   CT = THRUST/(0.5*RHO*U^2*S);


% Lift curve slope of the 3D airplane
% -----------------------------------

   % cLalpha of wing
        CLAW =(57.3*CLA2DW)/(1 + CLA2DW*57.3/(pi*E1*AR));
        % cLalpha of tail
        CLAT =(57.3*CLA2DT)/(1 + CLA2DT*57.3/(pi*E1T*ART));
        % dEpsilon/dAlpha (downwash on tail due to alpha of wing)
        % Smetana p. 69, last equation

        DEDA = 20.0*(CLAW/57.3)*(((1/TR)^0.3)/AR^0.725)*(3.0*CH/LT1)^0.25;

        fprintf(fid1,'\n dEps/dAlpha wing effect on tail  = %4.6f ',DEDA);

        % approximate cLalpha of airplane by cLalpha of wing only
   CLA = CLAW;


% cDalpha (Smetana, pp. 64-65)

        % p. 65 last eq.
        % note: cd0alpha of airplane is approximated by cD0alpha of 2D airfoil

        CDA = CDA2DW + 2*CL*CLA/(pi*E*AR);


% cMalpha

        % contribution of wing (Smetana, p. 68, 1st equation)
        CMAW = CLA*((1+(2*CL/(pi*E*AR))*((ALPHA-IWING)/57.3)+CD/CLA)*(XA/CH)+...
        (2*CL/(pi*E*AR)-(ALPHA-IWING)/57.3-CL/CLA)*(ZA/CH));
        % contribution of tail (Smetana, p. 69 4th equation from top of page)
        CMAT = CLAT*(1 - DEDA)*(ST/S)*(LT/CH)*EFF;
        % for fuselage contribution, use Fig. 16 in Smetana (page 70)
        % check to make sure that (position of wing 1/4 root chord)/(length of body)
        % is within the range 0.08 to 0.65

        XLBo = XFUS/LB;

        if (XLBo < 0.08 | XLBo > 0.65)        % if loop:  Calculates KF, CMA, CMAFUS
```

```
            disp('XLBo = ');
            disp(XLBo);
            disp('ERROR: XLBo IS OUTSIDE DESIRABLE RANGE OF 0.1 TO 0.65');
            %disp('WHEN CALCULATING KF');
            %error('Program stopped');
            CMA = -0.2;
      else
            % Fig. 16, p. 70 Smetana
            KF = -19.6558*XLBo^4+50.3465*XLBo^3-26.5479*XLBo^2+7.47299*XLBo-0.464064;
            % Smetana p. 70, 1st Eq.
            % fuselage contribution
            CMAFUS = KF*WFUS*WFUS*LB/(S*CH);

            % complete cMalpha (Smetan, p. 74)

            CMA = CMAW + CMAFUS - CMAT;
      end
% end if loop:  Calculates KF, CMA, CMAFUS

% cL-alphadot (Smetan, p. 76)
% ---------

    CLDA = 2*CLAT*DEDA*(LT1/CH)*(ST/S)*EFF;

% cD-alphadot (Smetana, p. 77)
% ----------------


    CDDA = 0.0;


% cM-alphadot (Smetan, p. 81)


  CMDA = -2*CLAT*DEDA*(LT/CH)*(LT1/CH)*(ST/S)*EFF;


% cL-q (Smetana, p. 85)
% ----------------------

  % orginial eq. in Smetana
      CLQ = 2.0*(XCG/CH)*CLA+2*(LT/CH)*CLAT*(ST/S)*EFF;
      % modified eq. (eliminate wing contribution) - not used currently
      %     CLQ=2.0*(LT/CH)*CLAT*(ST/S)*EFF

% cD-q (Smetana, p. 86)
% ---------------------

  CDQ = 0.0;

% cM-q (Smetana, p. 88)
% ----------------------

      % orginial eq. in Smetana
      CMQ =-2*(XCG/CH^2)*abs(XCG)*CLA - 2*(LT/CH)^2*CLAT*(ST/S)*EFF;
      % modified eq. (eliminate wing contribution) not used currently
      %     CMQ=-2.0*(LT/CH)**2*CLAT*(ST/S)*EFF

% cL-deltaE (Smetana, pp. 89-94)

      % Fig, 20, p. 90: ratio of elevator area to tail area

      CHECHT = CHE/CHT;
```

```
% check it is between 0 and 1
if (CHECHT < 0 | CHECHT > 1)
disp('CHE/CHT = ');
disp(CHECHT);
disp('ERROR: CHE/CHT IS OUTSIDE DESIRABLE RANGE OF 0.0 TO 1.0');
disp('WHEN CALCULATING CLDE2D');
%error('Program stopped');
end

% Fig. 21, p. 91 Smetana: Find the 2D dcL/dcDE for the tail's
% airfoil / flap combination for different flap chord to airfoil
% chord ratios (between 0 and 1)
% Actually, there should be a check here to make sure CHE/CHT
% is within range.

% chord ratio elevator / c

cratioT = CHE/CHT;

fprintf(fid1,'\n elvator chord : tail chord ratio = %4.6f ',cratioT);

% Note: the equation used here by Smetana fits the curve in Fig. 21
% denoted "High aspect ratios".
% For tail surfaces it might be more apppropriate to use the lower curve
% for "Low aspect ratios".
% Here the "high ASpect Ratio curve is used when ARtail is greater than 4.5
% otherwise the "low aspect ratio curve is used

% when CHE/CHT is less than 0.4, aspect ratio does not matter

if (CHE/CHT <= 0.4)   % if loop for calc of CLDE2D
CLDE2D = -0.0580767*(CHE/CHT)^4+0.146101*(CHE/CHT)^3-0.169823*...
(CHE/CHT)^2+0.194084*(CHE/CHT)-0.00202833;

        % now if CHE/CHT is greater than 0.4, in the case ofhigh ARt:
elseif (ART > 4.5)
CLDE2D = -0.0580767*(CHE/CHT)^4+0.146101*(CHE/CHT)^3-0.169823*...
(CHE/CHT)^2+0.194084*(CHE/CHT)-0.00202833;
else
        % when CHE/CLT is greater than 4.5 and ARt is low:
        % (we use a simple linear approximation of the low aspect ratio
        % curve above CHE/CHT of 0.4)
        CLDE2D = 0.0563 + 0.0321*(CHE/CHT-0.4);
end                                          % end if loop for calc of CLDE2D

% dcl/ddelE 2D
fprintf(fid1,'\n dcl / ddeltaE Tail 2D           = %4.6f ',CLDE2D);

% check to make sure that ARtail is between 0 and 10
% for Fig. 23 p. 93

if (ART < 0 | ART > 10)
disp('ART = ');
disp(ART);
disp('ERROR: ART IS OUTSIDE DESIRABLE RANGE OF 0.0 TO 10.0');
disp('WHEN CALCULATING CLDE');
%error('Program stopped');
end
% Smetana, Fig. 93 upper right small figure

AD2D = 4.65842*(CHE/CHT)^4-10.9873*(CHE/CHT)^3+9.47521*(CHE/CHT)^2-...
4.09969*(CHE/CHT)-0.0432959;
```

```
        fprintf(fid1,'\n ce/ctail                          = %4.6f ',CHE/CHT);
        fprintf(fid1,'\n AlphaDelta sub cl                 = %4.6f ',AD2D);

        % Now, depending on the result from the upper right small figure,
        % enter the main figure 23 and use the relevant curve

        if (AD2D >= -0.1)

       AD = -0.0000580764*ART^5+0.00203746*ART^4-0.0284903*ART^3+0.203783*ART^2-...
          0.802715*ART+2.77199;

    elseif (AD2D < -0.1 & AD2D >= -0.2)

       AD1 = -0.0000580764*ART^5+0.00203746*ART^4-0.0284903*ART^3+0.203783*ART^2-...
                0.802715*ART+2.77199;
                AD2 = -0.000858209*ART^3+0.0224724*ART^2-0.206709*ART+1.81719;
                AD = AD1 + (AD2 - AD1)*(AD2D -(-0.1))/(-0.1);

    elseif (AD2D < -0.2 & AD2D >= -0.3)

       AD1 = -0.000858209*ART^3+0.0224724*ART^2-0.206709*ART+1.81719;
                AD2 = 0.000132994*ART^4-0.00394386*ART^3+0.0450323*ART^2-0.249244*ART+...
                1.69816;
       AD = AD1 + (AD2 - AD1)*(AD2D -(-0.2))/(-0.1);

    elseif (AD2D < -0.3 & AD2D >= -0.4)

       AD1 = 0.000132994*ART^4-0.00394386*ART^3+0.0450323*ART^2-0.245244*ART+1.69816;
                AD2 = -0.00000724351*ART^6+0.000235239*ART^5-
0.00285365*ART^4+0.0150882*ART^3-...
                0.0214656*ART^2-0.108685*ART+1.47429;
       AD = AD1 + (AD2 - AD1)*(AD2D -(-0.3))/(-0.1);

    elseif (AD2D < -0.4 & AD2D >= -0.5)

       AD1 = -0.00000724351*ART^6+0.000235239*ART^5-0.00285365*ART^4+0.0150882*ART^3-...
                0.0214656*ART^2-0.108685*ART+1.47429;
                AD2 = -0.000011681*ART^5+0.000430597*ART^4-
0.00621814*ART^3+0.0456723*ART^2-...
                0.184917*ART+1.41226;
                AD = AD1 + (AD2 - AD1)*(AD2D -(-0.4))/(-0.1);

    elseif (AD2D < -0.5 & AD2D >= -0.6)

       AD1 = -0.000011681*ART^5+0.000430597*ART^4-0.00621814*ART^3+0.0456723*ART^2-...
                0.184917*ART+1.41226;
                AD2 = -0.0000073708*ART^5+0.000307952*ART^4-
0.00489224*ART^3+0.0380384*ART^2-...
                0.154526*ART+1.32116;
       AD = AD1 + (AD2 - AD1)*(AD2D -(-0.5))/(-0.1);

    elseif (AD2D < -0.6 & AD2D >= -0.7)

       AD1 = -0.0000073708*ART^5+0.000307952*ART^4-0.00489224*ART^3+0.0380384*ART^2-...
                0.154526*ART+1.32116;
                AD2 = 0.00000243494*ART^4-0.000255415*ART^3+0.0057768*ART^2-
0.0487234*ART+1.16569;
                AD = AD1 + (AD2 - AD1)*(AD2D -(-0.6))/(-0.1);

        elseif (AD2D < -0.7 & AD2D >= -0.8)

        AD1 = 0.00000243494*ART^4-0.000255415*ART^3+0.0057768*ART^2-0.0487234*ART+1.16569;
```

```
                AD2 = 0.0000255244*ART^4-0.00080707*ART^3+0.00935254*ART^2-
0.0487466*ART+1.12006;
                AD = AD1 + (AD2 - AD1)*(AD2D -(-0.7))/(-0.1);

   else

        AD1 = 0.0000255224*ART^4-0.00080707*ART^3+0.00935254*ART^2-0.0487466*ART+...
          1.12006;
        AD2 = 1.0;
        AD = AD1 + (AD2 - AD1)*(AD2D -(-0.8))/(-0.2);

        end

        fprintf(fid1,'\n (alphaDelta cL)/(alphaDelta cL)  = %4.6f ',AD);

        % cL-DeltaE (Smetana, p. 94, top equation

        CLDE = CLDE2D*(CLAT/(CLA2DT))*AD*(ST/S)*EFF;

        % Modified
        %SSE = 2*CHE*BE;
        %CLDE = CLDE*2*BE/BT;

        % store calculated cL-deltaE in CLIN

        CLIN = CLDE;


% store  cL-deltaE in CLIN

CLIN = CLDE;

% cD-deltaE
% ---------

        % calculation assumes it is 0
        % however, Smetana pp. 95-100 can be used too.

        CDDE = 0.0;
        % store in CDIN
        CDIN = CDDE;

% if CDDE is given, store in CDIN
CDIN = CDDE;

% CM-deltaE
% -------------
        % p. 101, eq. at bottom of page

        CMDE = (-LT/CH)*CLDE;
        % store in CMIN
        CMIN = CMDE;

% if CMDE is given, store in CMIN
CMIN = CMDE;

% cL-u
% ------

   CLU = 0.0;

% cD-u
%------
```

```
   CDU = 0.0;

% cM-u
%------

   CMU = 0.0;

% cT-u
% ----

   CTU = 0.0;

% cT-rpm
% ----------

   CTRPM = 0.0;


% Estimate Static Margin

StaticMargin = -CMA/CLA;

% Write Output File

fprintf(fid1,'\n Flight Conditions') ;
fprintf(fid1,'\n -----------------');
fprintf(fid1,'\n Density, slug/ft^3      = %4.6f',RHO);
fprintf(fid1,'\n Velocity, ft/s          = %4.2f',U);
fprintf(fid1,'\n Mass, slugs             = %4.2f',MS);
fprintf(fid1,'\n Iyy, slug*ft^2          = %4.2f',IYY);
fprintf(fid1,'\n Thrust, lbs             = %4.2f',THRUST);
fprintf(fid1,'\n Vertical distance to thrust line, ft     = %4.2f ',ZJ);
fprintf(fid1,'\n G*Cos(GAMMA), ft/sec^2  = %4.2f',GCOSGM);
fprintf(fid1,'\n G*Sin(GAMMA), ft/sec^2  = %4.2f',GSINGM);
fprintf(fid1,'\n COSXz                   = %4.2f ',COSXZ);
fprintf(fid1,'\n SINXz                   = %4.2f ',SINXZ);
fprintf(fid1,'\n');
fprintf(fid1,'\n');
fprintf(fid1,'\n Aircraft Characteristic Values');
fprintf(fid1,'\n ------------------------------');
fprintf(fid1,'\n Wing Area, ft^2                    = %4.2f',S);
fprintf(fid1,'\n Horizontal Tail Area, ft^2         = %4.2f',ST);
fprintf(fid1,'\n Wing Span, ft                      = %4.2f',B);
fprintf(fid1,'\n Horizontal Tail Span, ft           = %4.2f',BT);
fprintf(fid1,'\n Wing Chord, ft                     = %4.2f',CH);
fprintf(fid1,'\n Horizontal Tail Chord, ft          = %4.2f',CHT);
fprintf(fid1,'\n Wing Aspect Ratio                  = %4.2f',AR);
fprintf(fid1,'\n Horizontal Tail Aspect Ratio       = %4.2f',ART);
fprintf(fid1,'\n Wing Taper Ratio                   = %4.2f',TR);
fprintf(fid1,'\n Horizontal Tail Taper Ratio        = %4.2f',TRT);
fprintf(fid1,'\n Wing AOA (Alpha), deg              = %4.2f',ALPHA);
fprintf(fid1,'\n Tail AOA (Alpha), deg              = %4.2f',ALPHAT);
fprintf(fid1,'\n Wing Incidence Angle, deg          = %4.2f',IWING);
fprintf(fid1,'\n Tail Incidence Angle, deg          = %4.2f',ITAIL);
fprintf(fid1,'\n');
fprintf(fid1,'\n');
fprintf(fid1,'\n Downwash Angle, deg                = %4.2f',DW);
fprintf(fid1,'\n Downwash/Alpha                     = %4.2f',DEDA);
fprintf(fid1,'\n Elevator Angle, deg                = %4.2f',DELTAE);
fprintf(fid1,'\n Elevator Area, ft^2                = %4.2f',SELEV);
fprintf(fid1,'\n Tail efficiency                    = %4.2f',EFF);
fprintf(fid1,'\n Elevator Chord, ft                 = %4.2f',CHE);
```

```
fprintf(fid1,'\n 2-D Wing CL-alpha                    = %4.2f',CLA2DW);
fprintf(fid1,'\n 2-D Tail CL-alpha                    = %4.2f',CLA2DT);
fprintf(fid1,'\n Zero-lift Drag Coeff.                = %4.2f',CD0) ;
fprintf(fid1,'\n 2-D Wing CD-alpha                    = %4.2f',CDA2DW);
fprintf(fid1,'\n');
fprintf(fid1,'\n');
fprintf(fid1,'\n Length of Fuselage, ft               = %4.2f',LB);
fprintf(fid1,'\n Max Width of Fuselage, ft            = %4.2f',WFUS);
fprintf(fid1,'\n C.G. to Tail Quarter-chord, ft       = %4.2f',LT);
fprintf(fid1,'\n Wing to Tail Quarter-chord, ft       = %4.2f',LT1);
fprintf(fid1,'\n C.G. to Wing A.C. (Chordwise), ft    = %4.2f',XA);
fprintf(fid1,'\n C.G. to Wing A.C. (Vertical), ft     = %4.2f',ZA);
fprintf(fid1,'\n Nose to Wing Quarter-chord, ft       = %4.2f',XFUS);
fprintf(fid1,'\n C.G. to Wing Quarter-chord, ft       = %4.2f',XCG);
fprintf(fid1,'\n C.G. to Tail (Vertical), ft          = %4.2f',ZT);
fprintf(fid1,'\n');
fprintf(fid1,'\n');
fprintf(fid1,'\n *****Longitudinal Stability Derivatives *****');
fprintf(fid1,'\n CL    = %4.6f',CL);
fprintf(fid1,'\n CLA   = %4.6f',CLA);
fprintf(fid1,'\n CLDA  = %4.6f',CLDA);
fprintf(fid1,'\n CLQ   = %4.6f',CLQ);
fprintf(fid1,'\n CLDE  = %4.6f',CLDE);
fprintf(fid1,'\n CLU   = %4.6f',CLU);
fprintf(fid1,'\n CT    = %4.6f',CT);
fprintf(fid1,'\n CD    = %4.6f',CD);
fprintf(fid1,'\n CDA   = %4.6f',CDA);
fprintf(fid1,'\n CDDA  = %4.6f',CDDA);
fprintf(fid1,'\n CDQ   = %4.6f',CDQ);
fprintf(fid1,'\n CDDE  = %4.6f',CDDE);
fprintf(fid1,'\n CDU   = %4.6f',CDU);
fprintf(fid1,'\n CTU   = %4.6f',CTU);
fprintf(fid1,'\n CM    = %4.6f',CM);
fprintf(fid1,'\n CMA   = %4.6f',CMA);
fprintf(fid1,'\n CMDA  = %4.6f',CMDA);
fprintf(fid1,'\n CMQ   = %4.6f',CMQ);
fprintf(fid1,'\n CMDE  = %4.6f',CMDE);
fprintf(fid1,'\n CMU   = %4.6f',CMU);
fprintf(fid1,'\n CTRPM = %4.6f',CTRPM);
fprintf(fid1,'\n');

% Notes:

%     CLIN, CDIN, AND CMIN ARE THE STABILITY DERIVATIVES DUE TO CONTROL
%     SURFACE DEFLECTIONS.  CLIN IS EITHER THE PARTIAL OF CL WITH
%     RESPECT TO ELEVATOR DEFLECTION OR FLAP DEFLECTION.  THE VALUE OF K
%     DETERMINES WHETHER IT IS CONCERNED WITH FLAPS OR ELEVATOR.

%     IF K IS GIVEN THE VALUE 1 THEN CLIN, CDIN, AND CMIN ARE PARTIAL
%     DERIVATIVES WITH RESPECT TO ELEVATOR DEFLECTION. IF K=2 THEN THE
%     PARTIALS ARE TAKEN WITH RESPECT TO FLAP DEFLECTION.

if (K == 1)
   fprintf(fid1,'\n deltaE Represents ELEVATOR DEFLECTION');
else
   fprintf(fid1,'\n deltaE Represents FLAP DEFLECTION');
end

fprintf(fid1,'\n CLIN   = %4.6f',CLIN);
fprintf(fid1,'\n CDIN   = %4.6f',CDIN);
fprintf(fid1,'\n CMIN   = %4.6f',CMIN);
fprintf(fid1,'\n K      = %4.6f',K);
fprintf(fid1,'\n');
```

```
%      CALCULATION OF DIMENSIONAL STABILITY DERIVATIVES
% ------------------------------------------------------
%      D AND DC ARE JUST CONSTANTS USED TO CALCULATE THE DIMENSIONAL
%      STABILITY DERIVATIVES.

D = RHO*U*S/MS;
DC = RHO*U*S*CH/IYY;

%      DIMENSIONAL STABILITY DERIVATIVES

       XU = D*(-(CDU+CD));
       ZU = D*(-(CLU+CL));
       MU = DC*(CMU+CM);
       TU = D*(CTU+CT);
       XW = (D/2)*(CL - CDA);
       ZW = (D/2)*(-(CLA + CD));
       MW = (DC/2)*CMA;
       ZDW = -RHO*S*CH*CLDA/(4*MS);
       XDW = -RHO*S*CH*CDDA/(4*MS);
       MDW = RHO*S*CH*CH*CMDA/(4*IYY);
       XQ = -RHO*U*S*CH*CDQ/(4*MS);
       ZQ = -RHO*U*S*CH*CLQ/(4*MS);
       MQ = DC*CH*CMQ/4;
       TRPM = 30*CH*D*CTRPM;
       XIN = -D*CDIN*U/2;
       ZIN = -D*CLIN*U/2;
       MIN = DC*CMIN*U/2;

       % The following calculations were added to the original Smetana code by
       % Doug Galloway on 29 Mar 00.  Equations are from Schmidt Table 4.1.
       ZAL = ZW*U;
       XAL = XW*U;
       MAL = MW*U;
       ZADOT = ZDW*U;
       XADOT = XDW*U;
       MADOT = MDW*U;

% Print the Dimensional Stability Derivatives

fprintf(fid1,'\n Dimensional Stability Derivatives ');
fprintf(fid1,'\n ---------------------------------');
fprintf(fid1,'\n XU     = %10.5f',XU);
fprintf(fid1,'\n ZU     = %10.5f',ZU);
fprintf(fid1,'\n MU     = %10.5f',MU);
fprintf(fid1,'\n TU     = %10.5f',TU);
fprintf(fid1,'\n XW     = %10.5f',XW);
fprintf(fid1,'\n ZW     = %10.5f',ZW);
fprintf(fid1,'\n MW     = %10.5f',MW);
fprintf(fid1,'\n XAL    = %10.5f',XAL);
fprintf(fid1,'\n ZAL    = %10.5f',ZAL);
fprintf(fid1,'\n MAL    = %10.5f',MAL);
fprintf(fid1,'\n XWD    = %10.5f',XDW);
fprintf(fid1,'\n ZWD    = %10.5f',ZDW);
fprintf(fid1,'\n MWD    = %10.5f',MDW);
fprintf(fid1,'\n XADOT  = %10.5f',XADOT);
fprintf(fid1,'\n ZADOT  = %10.5f',ZADOT);
fprintf(fid1,'\n MADOT  = %10.5f',MADOT);
fprintf(fid1,'\n XQ     = %10.5f',XQ);
fprintf(fid1,'\n ZQ     = %10.5f',ZQ);
fprintf(fid1,'\n MQ     = %10.5f',MQ);
fprintf(fid1,'\n TRPM   = %10.5f',TRPM);
fprintf(fid1,'\n XIN    = %10.5f',XIN);
```

```
fprintf(fid1,'\n ZIN     = %10.5f',ZIN);
fprintf(fid1,'\n MIN     = %10.5f',MIN);
```

C.22 VdocLongDynm.m

This file determines the longitudinal dynamic modes of motion. It is called by *analyze.m*.

```
%*************************************************************************
%  April 19, 2000
%-------------------------------------------------------------------------
%  VdocLongDynm.m     Longitudinal Dynamic Stability
%-------------------------------------------------------------------------
%
% For the Longitudinal Case this program calculates:
%       1) Pertinent specifications for each characteristic mode
%                       - Short Period
%                       - Long Period
%       2) Time response to control inputs
%
% This is achieved by constructing the governing state equations in
% matrix form for the lateral-directional equations of motion
global A
global Bo

% Variables changed to match the VdocLongStab.m
V = U;
g = GRAVIT;
ao = ALPHA; %ALPHA is in degrees
ao = ao*pi/180;

%Components of X vector
u = 0;
a = 0;
q = 0;
theta = 0;

%Create Matrix Equations
Xo = [(u/V) a q theta];
Xo = transpose(Xo);

De = [1];       %Elevator input
A = zeros(4,4);
Bo = zeros(4,1);

A(1,1) = XU;
A(1,2) = XAL/V;
A(1,3) = 0;
A(1,4) = -(g/V)*cos(ao);
A(2,1) = V*ZU/(V-ZADOT);
A(2,2) = ZAL/(V-ZADOT);
A(2,3) =(V+ZQ)/(V-ZADOT);
A(2,4) = -g*sin(ao)/(V-ZADOT);
A(3,1) = V*MU + MADOT*V*ZU/(V-ZADOT);
A(3,2) = MAL +MADOT*ZAL/(V-ZADOT);
A(3,3) = MQ+MADOT*(V+ZQ)/(V-ZADOT);
A(3,4) = 0;
A(4,1) = 0;
```

```
A(4,2) = 0;
A(4,3) = 1;
A(4,4) = 0;

%Control Vector
Bo(1,1) = XIN/V;
Bo(2,1) = ZIN/(V-ZADOT);
Bo(3,1) = MIN + MADOT*ZIN/(V-ZADOT);
Bo(4,1) = 0;
t = 0.0:0.10:100.0;
Co = ones(1,4);
D = 1;

if S < 0
A = ones(4,4);
end

%Determine the characteristic roots of A
LongModes = eig(A);
%Determine natural frequency and damping ratio for each mode
[Wn1,Z1] = damp(LongModes);  %Reference Schmide pg. 239

SPr1 = LongModes(1);
SPr2 = LongModes(2);
SPdamp = Z1(1);
SPnat = Wn1(1);
Phugr1 = LongModes(3);
Phugr2 = LongModes(4);
Phugdamp = Z1(3);
Phugnat = Wn1(3);

%if (imag(SPr1)==0)| (imag(SPr2)==0)
        SPnat = sqrt(-MAL);
        SPdamp = -(MADOT+MQ)/(2*SPnat);
%end

%if (imag(Phugr1)==0)| (imag(Phugr2)==0)
        Phugnat = sqrt(2)*GRAVIT/U;
        Phugdamp = 1/(sqrt(2)*CL/CD);
%end


if (imag(A) == zeros(4,4))
   if(imag(Bo) == zeros(4,1))
        % Model response to a unit impulse
        [Y1,X1,t] = impulse(A,Bo,Co,D,De,t);

        % Model response to a step input
        [Y2,X2,t] = step(A,Bo,Co,D,De,t);

        % Plot Dynamic Response
        %figure;
        x1a = X1(:,1);
        x1b = X1(:,2);
        x1c = X1(:,4);
        %subplot(2,1,1);
        %plot(t,x1a,t,x1b,t,x1c)%normalized velocity
        %xlabel('time');
        %ylabel('Perturbation value ');
        %legend('u/U (-)','AOA (deg)','Theta (deg)');
        %title('Longitudinal Response to an Elevator Impulse Input');

        %figure;
```

```
        x2a = X2(:,1);
        x2b = X2(:,2);
        x2c = X2(:,4);
        %subplot(2,1,2);
        %plot(t,x2a,t,x2b,t,x2c)%normalized velocity
        %xlabel('time');
        %ylabel('Perturbation value ');
        %legend('u/U (-)','AOA (deg)','Theta (deg)');
        %title('Longitudinal Response to an Elevator Step Input');
    end
end
SPr1print = [real(SPr1);imag(SPr1)];
SPr2print = [real(SPr2);imag(SPr2)];
Phug1print = [real(Phugr1);imag(Phugr1)];
Phug2print = [real(Phugr2);imag(Phugr2)];
fprintf(fid1,'\n');
fprintf(fid1,'\n');
fprintf(fid1,'\n Longitudinal Dynamic Stability Results ');
fprintf(fid1,'\n ---------------------------------------');
fprintf(fid1,'\n Short Period roots:  Real          Imaginery');
fprintf(fid1,'\n                      %4.5f       %4.5f',SPr1print);
fprintf(fid1,'\n                      %4.5f       %4.5f',SPr2print);
fprintf(fid1,'\n Short Period damp ratio = %4.5f',SPdamp);
fprintf(fid1,'\n Short Period nat. freq. = %4.5f',SPnat);
fprintf(fid1,'\n Phugoid roots:       Real          Imaginery');
fprintf(fid1,'\n                      %4.5f       %4.5f',Phug1print);
fprintf(fid1,'\n                      %4.5f       %4.5f',Phug2print);
fprintf(fid1,'\n Phugoid damping ratio   = %4.5f',Phugdamp);
fprintf(fid1,'\n Phugoid natural freq.   = %4.5f',Phugnat);
```

C.23 VdocLatDir.m

This file is the original Smetana code with some small modifications. It determines the lateral-directional stability derivatives and is called by *analyze.m*.

```
%************************************************************************
%  April 19, 2000
%------------------------------------------------------------------------
%  VdocLatDir.m     Lateral/Directional Stability Derivatives
%------------------------------------------------------------------------
%
% For the Lateral-Directional Case this program calculates:
%       1) Non-Dimensional Stability Derivatives
%       2) Dimensional Stability Derivatives
%
% Written by Prof. Frederick Smetana
%
% Derivation of Equations of Motion is based on
% 'Dynamics of the Airframe', Bureau of Aeronautics Report, AE-61-4II.
%
%    ASSUMPTIONS:
%
%       1) THE AIRFRAME IS ASSUMED TO BE A RIGID BODY.
%       2) THE EARTH IS ASSUMED TO BE FIXED IN SPACE, AND, UNLESS
%          SPECIFICALLY STATED OTHERWISE, THE EARTH'S ATMOSPHERE IS
%          ASSUMED TO BE FIXED WITH RESPECT TO THE EARTH.
```

```
%       3) THE MASS OF THE AIRPLANE IS ASSUMED TO REMAIN CONSTANT FOR
%          THE DURATION OF ANY PARTICULAR DYNAMIC ANALYSIS.
%       4) THE X-Z PLANE IS ASSUMED TO BE A PLANE OF SYMMETRY.
%       5) THE DISTURBANCES FROM THE STEADY FLIGHT CONDITION ARE ASSUMED
%          TO BE SMALL ENOUGH SO THAT THE PRODUCTS AND SQUARES OF THE
%          CHANGES IN VELOCITIES ARE NEGLIGIBLE IN COMPARISON WITH THE
%          CHANGES THEMSELVES.  ALSO, THE DISTURBANCE ANGLES ARE ASSUMED
%          TO BE SMALL ENOUGH SO THAT THE SINES OF THESE ANGLES MAY BE
%          SET EQUAL TO THE ANGLES AND THE COSINES SET EQUAL TO ONE.
%          PRODUCTS OF THESE ANGLES ARE ALSO APPROXIMATELY ZERO AND CAN
%          BE NEGLECTED.  AND, SINCE THE DISTURBANCES ARE SMALL, THE
%          CHANGE IN AIR DENSITY ENCOUNTERED BY THE AIRPLANE DURING ANY
%          DISTURBANCE CAN BE CONSIDERED TO BE ZERO.
%       6) DURING THE STEADY FLIGHT CONDITION, THE AIRPLANE IS ASSUMED
%          TO BE FLYING WITH WINGS LEVEL AND ALL COMPONENTS OF VELOCITY
%          ZERO EXCEPT U SUB 0.  W SUB 0 = 0 BECAUSE THE STABILITY AXES
%          WERE CHOSEN AS THE REFERENCE AXES.
%       7) THE FLOW IS ASSUMED TO BE QUASI-STEADY.
%
% The Pertinent Aircraft Characteristics are Defined as Follows:
%
% GRAVITY (Assume = 32.2 ft/s^2 for this Altitude Range) and
% GCOSGM and GSINGM are the Products of the Acceleration Due to
% GRAVITY (Assumed = 32.2 ft/s^2 for this Altitude Range) and the
% COSINE and SINE Respectively of the Initial Flight Path Angle,
% GAMMA, (Usually Zero for Level Flight).
%
% Input Data:
%
% Flight Conditions
%
% RHO - Density at altitude (slug/ft^3)
% U - Aircraft speed (ft/s)
%
% GAMMA - Flight Path Angle (Deg)
% G - acceleration of gravity
%
% CL - Airplane Lift Coefficient
% ALPHA - Angle of Attack (Deg)
% CD0 - Parasite Drag Coefficient
%
% Airplane geometry:
%
% C.G. Location:
%
% XM - Distance from the Nose to the Center of Gravity (ft)
%
% Wing:
%
% CROOT - Root Chord of the Wing (ft)
% B - Wing Span (ft)
% TR - Wing Taper Ratio (Tip Chord/Root Chord)
% SA - Wing Sweep Angle (Positive Aft, Deg)
% DIH - Wing Dihedral Angle (Positive Up, Deg)
%
% Wing Location:
%
% Xlewing - Distance from the Nose to the Leading Edge of the Wing (ft)
% ZW - Distance From Body Centerline to Quarter-Chord Point of
%              Exposed Wing Root Chord (Positive for Quarter-Chord Point Below
%              the Body Centerline, ft)
%
% 2D Aerodynamics:
```

```
%
% CLA2DW - Two-Dimensional Lift Curve Slope of the Wing (1/rad)
% CLA2DH - Two-Dimensional Lift Curve Slope of the Horizontal Tail (1/rad)
%
% Horizontal Tail:
%
% SAH - Sweep Angle of the Horizontal Tail (Deg)
% BH - Span of the Horizontal Tail (ft)
% Crtail - Root Chord of the Horizontal Tail (ft)
% TRH - Taper Ratio of the Horizontal Tail (Tip Chord/Root Chord)

%
% Horizontal Tail Location:
%
% Xletail - Distance from the Nose to the Leading Edge of the
%                              Horizontal Tail (ft)
%
% Vertical tail:
%
% BV - Span of the Vertical Tail (ft)
% Crvt - Root Chord of the Vertical Tail (ft)
% TRvt - Taper Ratio of the Vertical Tail (Tip Chord/Root Chord)
% ETAV - Efficiency Factor of the Vertical Tail
%
% Vertical Tail Location:
%
% Xlevt - Distance from the Nose to the Leading Edge of the Vertical Tail (ft)
% ZV - Distance from the Center of Pressure of the Vertical Tail to the
%          X-Axis (Positive for a Vertical Tail Above the X-axis, ft)
%
% Control Surfaces:
%
% Aileron:
% BA - Span of a single Aileron (ft)
% CA - Chord of an Aileron (ft)
% YI - Distance from the Body Centerline to the Inboard Edge of the Aileron (ft)
%
% Rudder:
% BR - Span of the Rudder (ft)
% CR - Chord of the Rudder (ft)
%
% Fuselage:
% Note: Fuselage Geometry is Defined for Different Figures
% in Smetana. Thus, there is redundancy in the definition
%
% Referring to Fig. 34 p. 114 Smetana:
%
% W - Maximum Width of the Fuselage (ft)
% LF - Length of the Fuselage (ft)
% H1 - Fuselage Height Measured at 1/4 LF from the Nose (ft)
% H2 - Fuselage Height Measured at 3/4 LF from the Nose (ft)
% H - Maximum Body Height at Wing-Body Intersection (ft)
% R1 - Radius of the Fuselage in the Vicinity of the Vertical Tail (ft)
%
% The Following Cards Define Variables Used to Approximate Fuselage
% Volume by Dividing the Volume into 4 Prismoids:
%
% HNOSE - Fuselage Height in the Nose Region (ft)
% WNOSE - Fuselage Width in the Nose Region (ft)
%
% HFCY - Fuselage Height at the Front of the Canopy (ft)
% WFCY - Fuselage Width at the Front of the Canopy (ft)
% LFCY - Length Along the Body Centerline from Nose to Front of Canopy (ft)
```

```
%
% LMH - Length Along the Body Centerline from Nose to Point of
%               Maximum Fuselage Height (ft)
%
% HBCY - Fuselage Height at the Back of the Canopy (ft)
% WBCY - Fuselage Width at the Back of the Canopy (ft)
% LBCY - Length Along the Body Centerline from Nose to Back of Canopy (ft)
%
% FUSELAGE AERODYNAMICS (Smetana p. 104)
%
% CLAFUS - cLalpha of fuselage (1/rad)
%          0.0525 for round fuselages
%          0.1243 for rectangular fuselages
%
% Airplane Inertial Characteristics
%
% IZZ - Moment of Inertia About the Z-Axis (slug-ft^2)
% IXZ - Product of Inertia (ft-lbs-sec^2)
% IXX - Moment of Inertia About the X-Axis (slug-ft^2)
% MS - Mass of the Airplane (slugs)

% Open Output File
fid1 = fopen('LatDirOut.txt','w');
fprintf (fid1, 'Solution for Smetana via MATLAB');
fprintf (fid1, '\n');

% Echo Check of Wing Input Data
fprintf(fid1,'\n Flight Condition') ;
fprintf(fid1,'\n Density, slug/ft^3      = %4.6f',RHO);
fprintf(fid1,'\n Velocity, ft/s          = %4.2f',U);
fprintf(fid1,'\n Flight Path Angle, rad  = %4.2f',GAMMA);
fprintf(fid1,'\n Gravity, ft/s^2         = %4.2f',GRAVIT);
fprintf(fid1,'\n Angle of Attack, deg    = %4.2f',ALPHA);
fprintf(fid1,'\n cL-alpha airplane       = %4.2f',CL);
fprintf(fid1,'\n CDo                     = %4.2f',CD0);
fprintf(fid1,'\n Mass, slugs             = %4.2f',MS);
fprintf(fid1,'\n Izz, slug*ft^2          = %4.2f',IZZ);
fprintf(fid1,'\n Ixz, slug*ft^2          = %4.2f',IXZ);
fprintf(fid1,'\n Ixx, slug*ft^2          = %4.2f',IXX);
fprintf(fid1,'\n');
fprintf(fid1,'\n Wing Data');
fprintf(fid1,'\n Wing Area, ft^2         = %4.2f',S);
fprintf(fid1,'\n Wing Root Chord, ft     = %4.2f',CROOT);
fprintf(fid1,'\n Wing Span, ft           = %4.2f',B);
fprintf(fid1,'\n Taper Ratio             = %4.2f',TR);
fprintf(fid1,'\n Wing Sweep Angle, rad       = %4.2f',SA);
fprintf(fid1,'\n Wing Dihedral, deg      = %4.2f',DIH);
fprintf(fid1,'\n x-coord wing a.c., ft   = %4.2f',XW);
fprintf(fid1,'\n ZW, ft                  = %4.2f',ZW);
fprintf(fid1,'\n cL-alpha Wing 2D        = %4.2f',CLA2DW);
fprintf(fid1,'\n') ;
fprintf(fid1,'\n Horizontal Tail Data');
fprintf(fid1,'\n Tail Area, ft^2         = %4.2f',SH);
fprintf(fid1,'\n Tail Root Chord, ft     = %4.2f',Crtail);
fprintf(fid1,'\n Tail Span, ft           = %4.2f',BH);
fprintf(fid1,'\n Taper Ratio             = %4.2f',TRH);
fprintf(fid1,'\n Tail Sweep Angle, rad   = %4.2f',SAH);
fprintf(fid1,'\n LT, ft                  = %4.2f',LT);
fprintf(fid1,'\n XHT, ft                 = %4.2f',XHT);
fprintf(fid1,'\n cL-alpha Horiz tail 2D  = %4.2f',CLA2DH);
fprintf(fid1,'\n') ;
fprintf(fid1,'\n Vertical Tail Data');
```

```
fprintf(fid1,'\n Tail Area, ft^2          = %4.2f',SV);
fprintf(fid1,'\n Tail Root Chord, ft      = %4.2f',Crvt);
fprintf(fid1,'\n Tail Span, ft            = %4.2f',BV);
fprintf(fid1,'\n XVT, ft                  = %4.2f',XVT);
fprintf(fid1,'\n ZV, ft                   = %4.2f',ZV);
fprintf(fid1,'\n Tail efficiency          = %4.2f',ETAV);
fprintf(fid1,'\n');
fprintf(fid1,'\n Aileron Data');
fprintf(fid1,'\n Aileron Span, ft         = %4.2f ',BA);
fprintf(fid1,'\n Aileron Chord, ft        = %4.2f ',CA);
fprintf(fid1,'\n YI, ft                   = %4.2f',YI);
fprintf(fid1,'\n');
fprintf(fid1,'\n Rudder Data');
fprintf(fid1,'\n Rudder Area, ft^2        = %4.2f ',SR);
fprintf(fid1,'\n Rudder Span, ft          = %4.2f',BR);
fprintf(fid1,'\n Rudder Chord, ft         = %4.2f',CR);
fprintf(fid1,'\n');
fprintf(fid1,'\n Fuselage Data');
fprintf(fid1,'\n Body length, ft          = %4.2f ',LF);
fprintf(fid1,'\n Max diameter, ft         = %4.2f ',WFUS);
fprintf(fid1,'\n H1, ft                   = %4.2f ',H1);
fprintf(fid1,'\n H2, ft                   = %4.2f ',H2);
fprintf(fid1,'\n H, ft                    = %4.2f ',H);
fprintf(fid1,'\n R1, ft                   = %4.2f ',R1);
fprintf(fid1,'\n HNOSE, ft                = %4.2f ',HNOSE);
fprintf(fid1,'\n WNOSE, ft                = %4.2f ',WNOSE);
fprintf(fid1,'\n HFCY, ft                 = %4.2f ',HFCY);
fprintf(fid1,'\n WFCY, ft                 = %4.2f ',WFCY);
fprintf(fid1,'\n LFCY, ft                 = %4.2f ',LFCY);
fprintf(fid1,'\n LMH, ft                  = %4.2f ',LMH);
fprintf(fid1,'\n HBCY, ft                 = %4.2f ',HBCY);
fprintf(fid1,'\n WBCY, ft                 = %4.2f ',WBCY);
fprintf(fid1,'\n LBCY, ft                 = %4.2f ',LBCY);
fprintf(fid1,'\n cLalpha fuselage         = %4.2f ',CLAFUS);
fprintf(fid1,'\n');

% ------------------
% START COMPUTATIONS
% ------------------

% Calculate LT,LV
LT = XHT-XM;
LV = XVT-XM;


fprintf(fid1,'\n');
□
fprintf(fid1,'\n ****Internal Calculations. Interim Results**** ');
□
fprintf(fid1,'\n Dist of Horiz Tail from c.g., ft        = %4.6f ',LT);
□
fprintf(fid1,'\n Dist of Vert Tail from c.g., ft         = %4.6f ',LV);

%  Determine Whether the Airplane is Low Wing (WP=3.0)
%  Mid Wing (WP=2.0), or High Wing (WP=1.0)

R = ZW/H;

if R >= .25
   WP = 3.0;
   fprintf(fid1,'\n Low Wing, WP = 3.0');
elseif R < -.25
   WP = 1.0;
```

```
   fprintf(fid1,'\n High Wing, WP = 1.0');
else
   WP = 2.0;
   fprintf(fid1,'\n Mid Wing, WP = 2.0');
end


% Calculate Effective Aspect Ratio of Vertical Tail and Lift Curve
% Slope of Vertical Tail

% Smetana, bottom of pg.113

AE = 1.55*BV*BV/SV;
if AE > 6.5
   %disp(' Vert Tail AR is outside [0,6.5] it is =');
   %disp(AE);
   %disp(' AE = 6.5 is used ');
   %AE = 6.5;
end

CLAVT =.0000397694*AE^5-.00069754*AE^4+.00461464*AE^3-.0158634*AE^2+...
   .0401979*AE+.00037328;

% Fig. 35 p. 114 in Smetana gives cLaVT in 1/deg. It has to be
% converted to 1/rad.
% Convert to 1/rad

CLAVT = CLAVT*57.2958;

fprintf(fid1,'\n Effective Aspect Ratio of Vert Tail     = %4.6f ',AE);
fprintf(fid1,'\n cLalpha of Vert Tail, 1/rad             = %4.6f ',CLAVT);

% Calculate Aspect Ratio of Wing and Horizontal Tail

AR = B*B/S;
ARH = BH*BH/SH;

fprintf(fid1,'\n Wing Aspect Ratio                       = %4.6f ',AR);
fprintf(fid1,'\n Horizontal Tail Aspect Ratio            = %4.6f ',ARH);

% Calculate the Mean Aerodynamic Chord
% In Smetana this is actually the Average Chord for the wing
% Here we use the conventional MAC
% mac  (P&H p. 90 2-78)

ctip = CROOT*TR;
CH = (CROOT+ctip-CROOT*ctip/(CROOT+ctip))*2/3;

fprintf(fid1,'\n Wing Mean Aerodynamic Chord, ft         = %4.6f ',CH);

% Estimate Body Side Area Using Four Trapezoids

SBS =(HNOSE+HFCY)*LFCY/2+(H+HFCY)*(LMH-LFCY)/2+(H+HBCY)*(LBCY-LMH)/2+...
   (HBCY+2*R1)*(LF-LBCY)/2;
SBS = abs(SBS);

fprintf(fid1,'\n Estimated Side Body Area, ft^2          = %4.6f ',SBS);

% Estimate Fuselage Volume Using Four Prismoids

V1 = LFCY*(2*(HNOSE*WNOSE+HFCY*WFCY)+HFCY*WNOSE+HNOSE*WFCY);
V2 = (LMH-LFCY)*(2*(HFCY*WFCY+H*WFUS)+H*WFCY+HFCY*WFUS);
V3 = (LBCY-LMH)*(2*(HBCY*WBCY+H*WFUS)+H*WBCY+WFUS*HBCY);
```

```
V4 = (LF-LBCY)*(2*(HBCY*WBCY+(2*R1)*(2*R1))+HBCY*2*R1+WBCY*2*R1);
FUSVOL = (V1+V2+V3+V4)/6;

fprintf(fid1,'\n Estimated Fuselage Volume                = %4.6f ',FUSVOL);

% The Following Sections Calculate the Individual Stability
% Derivative Values

% cYbeta (Smetana, pp. 103-107)       (1/rad)
% ------
fprintf(fid1,'\n');
fprintf(fid1,'\n Contributions to cYbeta');

% Wing contribution:
% Sweep
CYBWS = CL*CL*6*tan(SA)*sin(SA)/(pi*AR*(AR+4*cos(SA)));
fprintf(fid1,'\n Due to wing sweep, 1/rad                 = %4.6f ',CYBWS);

% Dihedral
CYBWD = -.0001*abs(DIH)*57.2958;
fprintf(fid1,'\n Due to wing dihedral, 1/rad              = %4.6f ',CYBWD);

% Fuselage
% Body Reference Area
BRA = FUSVOL^(2/3);

% Wing Fuselage Interference Factor (Fig. 30)
if ZW < 0
   KI = 1 +.485*(ZW*2/H);
else
   KI = 1 -.85555*(ZW*2/H);
end

CYBFUS = -KI*CLAFUS*BRA/S;
fprintf(fid1,'\n Due to the fuselage, 1/rad               = %4.6f ',CYBFUS);

% Vertical Tail
% Top of p. 107
SWDPR = .724 + 3.06*SV/(S*(1+cos(SA)))+.4*ZW/H+.009*AR;

% Fig. 31
PAR = BV/(2*R1);
KY = 1 +.166*(PAR-3.5);

if PAR <= 2
   KY = .75;
end
if PAR >= 3.5
   KY = 1;
end

CYBT = -KY*CLAVT*SWDPR*SV/S;
fprintf(fid1,'\n Due to the vertical tail, 1/rad          = %4.6f ',CYBT);

% Total cYbeta (1/rad)

CYB = CYBWS+CYBWD+CYBFUS+CYBT;

% clbeta (pp. 108-111)
% ------
fprintf(fid1,'\n');
fprintf(fid1,'\n Contributions to cLbeta');
```

```
ARR = AR;

% If AR is outside of the [1.5,10] range set it to the lower or upper
% limit depending on whether it is smaller than 1.5 or greater than 10.
       if AR < 1.5
      ARR = 1.5;
      disp('Value of AR used to calculate CLB is outside preferred');
      disp('range of 1.5 to 10.0');
      disp('AR =');
      disp(AR);
   elseif AR >10
      ARR = 10;
      disp('Value of AR used to calculate CLB is outside preferred');
      disp('range of 1.5 to 10.0');
      disp('AR =');
      disp(AR);
   end

% Fig. 33, p. 110

% Dihedral Effect

CLBOD =((-.20833-sqrt(.0434-4.6296*(1.0052-ARR)))/2.3148148+...
       .7*(1-TR)*(1-TR)*(ARR-1.5)/6.5)*.0001;

% Convert to Radians
CLBWD = CLBOD*DIH*57.2958;
fprintf(fid1,'\n Due to dihedral effect, 1/rad          = %4.6f ',CLBWD);

% Wing Contribution (p. 109)
CLBWWD = CL*(-1.25*(.71*TR+.29)/(ARR*TR)+.05);
fprintf(fid1,'\n Due to the wing, 1/rad                 = %4.6f ',CLBWWD);

% Combined Wing Effect:
CLBW = CLBWD+CLBWWD;
fprintf(fid1,'\n Due to combined wing effect, 1/rad     = %4.6f ',CYBT);

% Vertical Tail
CLBVT = -CLAVT*SV*ZV*ETAV/(S*B);
fprintf(fid1,'\n Due to the vertical tail               = %4.6f ',CLBVT);

% Table 10 p. 111
if WP ~= 1.0
   if WP ~= 2.0
      DCLB=.00064
   else
      DCLB = 0.0;
   end
else
   DCLB = -.00044;
end

% Total cLbeta
CLB = CLBW+CLBVT+DCLB;

% cNbeta
% ------

% Fig. 34, p. 114, and eq at center of p. 113

A = XM/LF;
if (A < 0.1) | (A > 0.8)
```

```
   % Print a warning message and set A to one of the extremes
   disp('Value of XM/LF used to calculate CNB is outside preferred');
   disp('range of 0.1 to 0.8');
   disp('XM/LF =');
   disp(A);

   if A < 0.1
      A = 0.1;
   end
   if A > 0.8
      A = 0.8;
   end
end
Xii =(A+.2)*10;
XP = Xii-3;

%  FIG. 37 on p. 117

R = LF*LF/SBS;
R = abs(R);

if (R < 2.5) | (R > 20)

       % Print a warning message
   disp('Value of LF*LF/SBS used to calculate CNB is outside preferred');
   disp('range of 2.5 to 20.0');
   disp('LF*LF/SBS =');
   disp(R);

       % Set R to one of the extremes (if it is out of range)
   if R < 2.5
      R = 2.5;
   end
   if R > 20
      R = 20;
   end
end

YO = -.0015609*R^3+.0675772*R^2-.999884*R+1.20213;
YP = .314*XP+YO;
Y = 12+YP;
YP = Y-8;

AA = sqrt(H1/H2);
if (AA < 0.8) | (AA > 1.65)

   % Print a warning message
       disp('Value of SQRT(h1/H2) used to calculate CNB is outside preferred');
   disp('range of 0.8 to 1.65');
   disp('SQRT(H1/H2) =');
   disp(AA);

   % Set AA to one of the extremes
   if AA < 0.8
      AA = 0.8;
   end
   if AA > 1.65
      AA = 1.65;
   end
end

XP = AA*YP;
```

```
AB = H/WFUS;
if (AB < .5) | (AB > 2.0)

% Print a warning message
   disp('Value of H/W used to calculate CNB is outside preferred');
   disp('range of 0.5 to 2.0');
   disp('H/W =');
   disp(AB);

   % Set AB to one of the extremes
   if AB < 0.5
      AB = 0.5;
   end
   if AB > 2
      AB = 2;
   end
end

YP = -((H/WFUS)^(.48*WFUS/H))*XP;
Y1 = YP;

% Reynolds number based on body length (Fig. 37 p. 117)
RHO1 = RHO*10000;

MU = .34777*RHO1+29.23345;
RN = RHO1*U*LF/(MU*.0001);
AC = RN*.000001;

if (AC < 0.0) | (AC > 80)

      % Print a warning message
   disp('Value of RN*10-6 used to calculate CNB is outside preferred');
   disp('range of 0.0 to 80.0');
   disp('RN*10-6 =');
   disp(AC);

   % Set AC to one of the extremes
   if AC < 0
      AC = 0
   end
   if AC > 80
      AC = 80;
   end
end

YP = AC/20;
Xii = .00833327*YP^3-.113571*YP^2+1.02095*YP+.446857;
B1 = Xii/2;
Xii = -2*(Y1-B1);
KB =(Xii-1)*.0005;

% Calculate cNbeta

CNB = -KB*SBS*LF*57.298/(S*B)-CYBT*LT/B;

% cLp
% ---
fprintf(fid1,'\n');
fprintf(fid1,'\n Contributions to cLp');

if (AR < 0) | (AR > 10)
      disp('Value of AR used to calculate CLP is outside preferred');
   disp('range of 0.0 to 10.0');
```

```
   disp('AR =');
   disp(AR);
end

CLPAO =(.00000929805*AR^5-.00031818*AR^4+.00379426*AR^3-.0157796*AR^2-...
   .0499261*AR-.0500059)+.209*(1-TR)^2*(AR-1)/9;

fprintf(fid1,'\n CLPAO, 1/rad                              = %4.6f ',CLPAO);
if(abs(SA) > 0.524)
   SA = 0.0;
end

CLPAOW = CLPAO*(AR+4*cos(SA))/((2*pi*AR/(CLA2DW*57.3)) + 4*cos(SA));
fprintf(fid1,'\n CLPAOW, 1/rad                             = %4.6f ',CLPAOW);

CLPW = CLPAOW-(CL*CL/(8*pi*AR*cos(SA)^2))*(1+2*sin(SA)^2*(AR+2*cos(SA))/...
   (AR+4*cos(SA)))-CD0/8;

fprintf(fid1,'\n Due to the wing, 1/rad                    = %4.6f ',CLPW);

CLPAOH = (.00000929805*ARH^5-.00031818*ARH^4+.00379426*ARH^3-.0157796*ARH^2-...
   .0499261*ARH-.0500059)+.209*(1-TRH)^2*(ARH-1)/9;

fprintf(fid1,'\n CLPAOH, 1/rad                             = %4.6f ',CLPAOH);

CLPH = ((0.5*SH*BH*BH)/(S*B*B))*(CLPAOH*(ARH+4*cos(SAH))/...
   ((2*pi*ARH/CLA2DH)+4*cos(SAH)));
fprintf(fid1,'\n Due to the horizontal tail, 1/rad         = %4.6f ',CLPH);

CLPV = 2*ZV*ZV*CYBT/(B*B);
fprintf(fid1,'\n Due to the vertical tail, 1/rad           = %4.6f ',CLPV);

CLP = CLPW+CLPH+CLPV;

% cYp (pp. 118-121)
% -----------------

if (SA < -.524) | (SA > .524)

%   disp('Value of SA used to calculate CYP is outside preferred');
%   disp('range of -.524 to .524 radians');
%   disp('SA =');
%   disp(SA);
end

if AR < 5
%   disp('Value of AR used to calculate CYP is is less than the suggested');
%   disp('minimum of 5.0; however, due due the negligible effect of this');
%   disp('derivative, one may ignore the effect of smaller AR on CYP.');
%   disp('AR =');
%   disp(AR);
end

Xii = -(SA*57.3-30)*7/60;
Y = Xii + 3*(1-TR^1.58495);
CYPOCL = (4-Y)/10;

if (DIH < -10) | (DIH > 10)
%   disp('Value of DIH used to calculate CYP is outside preferred');
%   disp('range of -10.0 to 10.0 degrees');
%   disp('DIH =');
%   disp(DIH);
end
```

```
P = (DIH-10)/17.5+.5;

%Calculate cYp
CYP = CYPOCL*CL+P*CLP;
% cNp
% ---
fprintf(fid1,'\n');
fprintf(fid1,'\n Contributions to cNp');

if (AR > 16)
%   disp('Value of AR used to calculate CNP is outside preferred');
%   disp('range of 0.0 to 16.0');
%   disp('AR =');
%   disp(AR);
end

CNPOCL = -.0000140468*AR^3+.000649352*AR*AR-.0122523*AR+.00192856;
CNPW = (CL*(AR+4)/(AR+4*cos(SA)))*(1+6*(1+cos(SA)/AR)*tan(SA)*tan(SA)/12)*CNPOCL;
fprintf(fid1,'\n Due to the wing, 1/rad                    = %4.6f ',CNPW);

P = ZV*2/B;

if (P < 0) | (P > .6)
%   disp('Value of P used to calculate CNP is outside preferred');
%   disp('range of 0.0 to 0.6');
%   disp('P =');
%   disp(P);
end

SIG1 = 22360.7*P^9-69947.1*P^8+93669.5*P^7-70229.7*P^6+32373.8*P^5-...
   9487.01*P^4+1764.41*P^3-200.25*P*P+11.9843*P-.00142786;

if (P >= .157)
   SIG2 = .533183*P^2-.78216*P+.35117;
else
   SIG2 = 216915*P^9-610122*P^8+724643*P^7-473060*P^6+185140*P^5-...
      44463.6*P^4+6444.43*P^3-531.134*P^2+21.1205*P+.0395735;
end

SIGMA1 = SIG2+TR*(SIG1-SIG2);
DELHOB = (ZV-(ZV*cos(ALPHA)-LT*sin(ALPHA)))/B;
SIGMA2 = 9.3*DELHOB*DELHOB*3/B;
ALPHA =ALPHA*pi/180;
CNPV = (1*CLAVT*SV*(ZV*sin(ALPHA)+LT*cos(ALPHA))/(S*B))*(2*(ZV*cos(ALPHA)-...
   LT*sin(ALPHA))/B-(SIGMA1+SIGMA2));
fprintf(fid1,'\n Due to the vertical tail, 1/rad            = %4.6f ',CNPV);
ALPHA=ALPHA*180/pi;
CNP = CNPW+CNPV;

% cYr
% ---
fprintf(fid1,'\n');
fprintf(fid1,'\n Contributions to cYr');

CYRW = .143*CL-.05;
fprintf(fid1,'\n Due to the wing, 1/rad                    = %4.6f ',CYRW);

CYRT = -2*LT*CYBT/B;
fprintf(fid1,'\n Due to the vertical tail, 1/rad           = %4.6f ',CYRT);

CYR = CYRW+CYRT;
```

```
% cLr
% ---
fprintf(fid1,'\n');
fprintf(fid1,'\n Contributions to cLr');

if (AR < 2) | (AR > 12)
   disp('Value of AR used to calculate CLR is outside preferred');
   disp('range of 2.0 to 12.0');
   disp('AR =');
   disp(AR);
end

YP = -.0000247674*AR^4+.00194154*AR^3-.0468052*AR^2+.456404*AR-.0980299+...
   2.15*(AR+32.4)*(TR^(1.12*(1-TR)))/44.4;
Y = YP+2;

if (SA < 0) | (SA > .524)
   disp('Value of SA used to calculate CLR is outside preferred');
   disp('range of 0.0 to 0.524 radians');
   disp('SA =');
   disp(SA);
end

Xii = 100*Y/(100-SA*57.3);
CLROCL = Xii/20;
CLRW = CLROCL*CL;
fprintf(fid1,'\n Due to the wing, 1/rad                  = %4.6f ',CLRW);

CLRT = -2*LT*ZV*CYBT/(B*B);
fprintf(fid1,'\n Due to the vertical tail, 1/rad         = %4.6f ',CLRT);

CLR = CLRW+CLRT;

% cNr
% ---
CNR = 2*LT*LT*CYBT/(B*B)-(.33*(1+3*TR)*CD0/(2+2*TR)-.02*(1-(AR-6)/13-...
   (1-TR)/2.5)*CL*CL);

% Stability derivatives due to aileron deflection
% cYdeltaA
% --------
CYDA = 0.0;

% cLdeltaA
% --------

if (AR < 4) | (AR > 12)
   %disp('Value of AR used to calculate CLDA is outside preferred');
   %disp('range of 4.0 to 12.0');
   %disp('AR =');
   %disp(AR);
end

K = 0;
P = YI*2/B;
CLDABT = 51.1702*P^8-238.294*P^7+451.162*P^6-444.985*P^5+241.797*P^4-...
   70.2549*P^3+10.5819*P^2-.408045*P+.0000119225;
CLDAAT = 2.0833*P^5-6.1553*P^4+5.36932*P^3-.567235*P^2+.170341*P-.000227237;

while K <= 1
   CLDABT = 51.1702*P^8-238.294*P^7+451.162*P^6-444.985*P^5+241.797*P^4-...
   70.2549*P^3+10.5819*P^2-.408045*P+.0000119225;
```

```
        CLDAAT = 2.0833*P^5-6.1553*P^4+5.36932*P^3-.567235*P^2+.170341*P-.000227237;

        if (AR > 6) & (AR < 10)
        CLDAOT = CLDABT+(CLDAAT-CLDABT)*(AR-6)/4;
        elseif AR <= 6
        CLDAOT = CLDABT;
        elseif AR >= 10
        CLDAOT = CLDAAT;
        end
    if K ~= 1
        CLDALT = CLDAOT;
       P = (YI+BA)*2/B;
    elseif K == 1
       CLDAOT = CLDAOT-CLDALT;
       R = CA/CH;
    end
    K = K+1;
end

if (R < 0) | (R > .4)
   disp('Value of CA/CH used to calculate CLDA is outside preferred');
   disp('range of 0.0 to 0.4');
   disp('CA/CH =');
   disp(R);
end

T = -65.1062*R^4+50.8227*R^3-15.7949*R^2+3.53383*R+.000043467;
CLdeltaA = CLDAOT*T;

% cNdeltaA
% --------
R = YI*2/B;

if (AR < 3) | (AR > 8)
   %disp('Value of AR used to calculate CNDA is outside preferred');
   %disp('range of 3.0 to 8.0');
   %disp('AR =');
   %disp(AR);
end

if AR < 3
       EF1 = -.36;
   EF2 = -.0889976*R^3-.00666109*R^2+.0419053*R-.284843;

elseif (AR >= 3) & (AR <= 4)
   AF1 = -.36;
   BF1 = .0538037*R^3-.133855*R^2+.00627854*R-.262321;
   EF1 = AF1+(AR-3)*(BF1-AF1);
   AF2 = -.0889976*R^3-.00666109*R^2+.0419053*R-.28483;
   BF2 = .0549387*R^3-.164298*R^2+.101864*R-.234861;
   EF2 = AF2+(AR-3)*(BF2-AF2);

elseif (AR > 4) & (AR < 6)
   AF1 = .0538037*R^3-.133855*R^2+.00627854*R-.262321;
   BF1 = -.0629823*R^3-.0173375*R^2-.0333427*R-.180026;
   EF1 = AF1+((AR-4)/2)*(BF1-AF1);
   AF2 = .0549387*R^3-.164298*R^2+.101864*R-.234861;
   BF2 = -.108911*R^3+.0291149*R^2+.0440961*R-.160312;
   EF2 = AF2+((AR-4)/2)*(BF2-AF2);

elseif (AR > 4) & (AR <= 8)
   AF1 = -.0629823*R^3-.0173375*R^2-.0333427*R-.180026;
   BF1 = -.110526*R^2+.013893*R-.146355;
```

```
   EF1 = AF1+((AR-6)/2)*(BF1-AF1);
   AF2 = -.108911*R^3+.0291149*R^2+.0440961*R-.160312;
   BF2 = -.191365*R^3+.147284*R^2-.00039296*R-.119884;
   EF2 = AF2+((AR-6)/2)*(BF2-AF2);

else
   EF1 = -.110526*R^2+.013893*R-.146355;
   EF2 = -.191365*R^3+.147284*R^2-.00039296*R-.119884;
end

EFF = EF2+((TR-.25)/.75)*(EF1-EF2);
CNDA = 2*EFF*CL*CLdeltaA;

% Stability derivatives due to rudder deflection
% cYdeltaR
% --------
R = SR/SV;

if (R < 0) | (R > .7)
   disp('Value of SR/SV used to calculate CYDR is outside preferred');
   disp('range of 0.0 to 0.7');
   disp('SR/SV =');
   disp(R);
end

T = 21.7949*R^5-46.4744*R^4+36.9347*R^3-14.259*R^2+3.70551*R-.000057815;
CYDR = CLAVT*T*SV/S;

% cLdeltaR
% --------

R = SR/SV;

if (R < 0) | (R > .7)
   disp('Value of SR/SV used to calculate CLDR is outside preferred');
   disp('range of 0.0 to 0.7');
   disp('SR/SV =');
   disp(R);
end

T = 21.7949*R^5-46.4744*R^4+36.9347*R^3-14.259*R^2+3.70551*R-.000057815;
CLDR = CLAVT*T*SV*ZV/(S*B);

%Modified
%CLDR = CLDR*BR/BV;


% cNdeltaR
% --------

R = SR/SV;

if (R < 0) | (R > .7)
   disp('Value of SR/SV used to calculate CNDR is outside preferred');
   disp('range of 0.0 to 0.7');
   disp('SR/SV =');
   disp(R);
end

T = 21.7949*R^5-46.4744*R^4+36.9347*R^3-14.259*R^2+3.70551*R-.000057815;

% Total cNdeltaR
```

```
CNDR = -CLAVT*T*SV*LT*ETAV/(S*B);

% Notes:

%     CLIN, CDIN, AND CMIN are the stability derivatives due to control
%     surface deflections.  CLIN is either the partial of CL with
%     respect to rudder deflection or aileron deflection.  The value of K
%     determines whether they refer to the rudder or ailerons.

%     If K is given the value 1 then CLIN, CDIN, AND CMIN are partial
%     derivatives with respect to rudder deflection. If K=2 then the
%     partials are taken with respect to flap deflection.

if (K == 1)
   CYIN = CYDR;
   CLIN = CLDR;
   CNIN = CNDR;
else
   CYIN = CYDA;
   CLIN = CLDA;
   CNIN = CNDA;
end


fprintf(fid1,'\n');
fprintf(fid1,'\n ******** SOLUTION ********');
fprintf(fid1,'\n ** Lateral Directional Stability Derivatives **');
fprintf(fid1,'\n cYbeta     = %4.6f',CYB);
fprintf(fid1,'\n cLbeta     = %4.6f',CLB);
fprintf(fid1,'\n cNbeta     = %4.6f',CNB);
fprintf(fid1,'\n cYp        = %4.6f',CYP);
fprintf(fid1,'\n cLp        = %4.6f',CLP);
fprintf(fid1,'\n cNp        = %4.6f',CNP);
fprintf(fid1,'\n cYr        = %4.6f',CYR);
fprintf(fid1,'\n cLr        = %4.6f',CLR);
fprintf(fid1,'\n cNr        = %4.6f',CNR);
fprintf(fid1,'\n cYdeltaA   = %4.6f',CYDA);
fprintf(fid1,'\n cLdeltaA   = %4.6f',CLdeltaA);
fprintf(fid1,'\n cNdeltaA   = %4.6f',CNDA);
fprintf(fid1,'\n cYdeltaR   = %4.6f',CYDR);
fprintf(fid1,'\n cLdeltaR   = %4.6f',CLDR);
fprintf(fid1,'\n cNdeltaR   = %4.6f',CNDR);


% end of aerodynamic stability derivative computation.

% Calculation of Dimensional Stability Derivatives
% Smetana pp. 34-36:

A = RHO*U*S;
F = RHO*U*S*B;
H = RHO*U*S*B*B;
D = RHO*U*U*S;
E = RHO*U*U*S*B;

% A, F, H, D, E are just constants used to calculate the dimesional
% stability derivatives

if  IZZ == 0
   IZZ = ICZZ;
end

YV = A*CYB/(2*MS);
```

```
YB = U*YV;
LV = F*CLB/(2*IXX);
LBe = U*LV;
NV = F*CNB/(2*IZZ);
NB = U*NV;
YP = F*CYP/(4*MS);
LP = H*CLP/(4*IXX);
NP = H*CNP/(4*IZZ);
YR = F*CYR/(4*MS);
LR = H*CLR/(4*IXX);
NR = H*CNR/(4*IZZ);
YDA = D*CYDA/(2*MS);
LDA = E*CLdeltaA/(2*IXX);
NDA = E*CNDA/(2*IZZ);
YDR = D*CYDR/(2*MS);
LDR = E*CLDR/(2*IXX);
NDR = E*CNDR/(2*IZZ);


% Print the Dimensional Stability Derivatives

fprintf(fid1,'\n Dimensional Stability Derivatives ');
fprintf(fid1,'\n ---------------------------------');
fprintf(fid1,'\n YV     = %10.5f',YV);
fprintf(fid1,'\n YB     = %10.5f',YB);
fprintf(fid1,'\n LV     = %10.5f',LV);
fprintf(fid1,'\n LB     = %10.5f',LBe);
fprintf(fid1,'\n NV     = %10.5f',NV);
fprintf(fid1,'\n NB     = %10.5f',NB);
fprintf(fid1,'\n YP     = %10.5f',YP);
fprintf(fid1,'\n LP     = %10.5f',LP);
fprintf(fid1,'\n NP     = %10.5f',NP);
fprintf(fid1,'\n YR     = %10.5f',YR);
fprintf(fid1,'\n LR     = %10.5f',LR);
fprintf(fid1,'\n NR     = %10.5f',NR);
fprintf(fid1,'\n YDA    = %10.5f',YDA);
fprintf(fid1,'\n LDA    = %10.5f',LDA);
fprintf(fid1,'\n NDA    = %10.5f',NDA);
fprintf(fid1,'\n YDR    = %10.5f',YDR);
fprintf(fid1,'\n LDR    = %10.5f',LDR);
fprintf(fid1,'\n NDR    = %10.5f',NDR);
```

C.24 VdocLatDynm.m

This file determines the lateral-directional dynamic modes of motion. It is called directly by *analyze.m*.

```
%*************************************************************************
%  April 19, 2000
%-------------------------------------------------------------------------
%  VdocLatDynm.m     Lateral-Directional Dynamic Stability
%-------------------------------------------------------------------------
%
% For the Lateral-Directional Case this program calculates:
%       1) Pertinent specifications for each characteristic mode
%          - Dutch Roll
```

```
%                       - Roll
%                       - Spiral
%       2) Time response to control inputs
%
% This is achieved by constructing the governing state equations in
% matrix form for the lateral-directional equations of motion
%
% Variables modified to match the output from VdocLatDir.m
V = U;
g = GRAVIT;
ao = ALPHA;             %ALPHA is in radians

% Calculate A matrix components
G = 1/(1-(IXZ^2/(IXX*IZZ)));
Lbp = G*(LBe + NB*(IXZ/IXX));
Lpp = G*(LP + NP*(IXZ/IXX));
Lrp = G*(LR + NR*(IXZ/IXX));
Nbp = G*(NB + LBe*(IXZ/IZZ));
Npp = G*(NP + LP*(IXZ/IZZ));
Nrp = G*(NR + LR*(IXZ/IZZ));

% Calculate Br and Ba vector components
Ldrp = G*(LDR + NDR*(IXZ/IXX));
Ndrp = G*(NDR + LDR*(IXZ/IZZ));
Ldap = G*(LDA + NDA*(IXZ/IXX));
Ndap = G*(NDA + LDA*(IXZ/IZZ));

%Components of X vector
beta = 0;
p = 0;
phi = 0;
r = 0;

%Create Matrix Equations
Xo = [beta p phi r];
Xo = transpose(Xo);
Xo = transpose(Xo);
Da = 1; %Aileron input
Dr = 1;  %Rudder input
Di = transpose([Da Dr]);
A(1,1) = YB/V;
A(1,2) = YP/V;
A(1,3) = (g/V)*cos(ao);
A(1,4) = (YR-V)/V;
A(2,1) = Lbp;
A(2,2) = Lpp;
A(2,3) = 0;
A(2,4) = Lrp;
A(3,1) = 0;
A(3,2) = 1;
A(3,3) = 0;
A(3,4) = 0;
A(4,1) = Nbp;
A(4,2) = Npp;
A(4,3) = 0;
A(4,4) = Nrp;
% Aileron Control Vector, Ba
Ba(1,1) = YDA/V;
Ba(2,1) = Ldap;
Ba(3,1) = 0;
Ba(4,1) = Ndap;
% Rudder Control Vector, Br
Br(1,1) = YDR/V;
```

```
Br(2,1) = Ldrp;
Br(3,1) = 0;
Br(4,1) = Ndrp;
%Bo = [Ba Br];

t = 0.0:0.10:100.0;
Co = ones(1,4);
D = 1;

%Determine the characteristic roots of A
LatModes = eig(A);

Rollroot = LatModes(3);
Troll = -1/Rollroot;

%Determine dutch roll characteristics
[Wn2,Z2] = damp(LatModes);

DRollr1 = LatModes(1);
DRollr2 = LatModes(2);

%disp('Dutch-roll damping ratio:');
DRdamp = Z2(2);

DRnat = Wn2(2);

%if (imag(DRollr1)==0)| (imag(DRollr2)==0)
        DRnat = sqrt(abs(NB));
        DRdamp = -0.5*NR/DRnat;
%end

Spiralroot = LatModes(4);

if (imag(A) == zeros(4,4))
   if (imag(Ba)==zeros(4,1))
     if (imag(Br)==zeros(4,1))
        %Calculate and plot response to aileron deflection
        % Model response to a step input
        [Y2,X2,t] = step(A,Ba,Co,D,Da,t);

        % Model response to a unit impulse
        [Y1,X1,t] = impulse(A,Ba,Co,D,Da,t);

        % Plot Dynamic Response to an Aileron Deflection
        %figure;
        x1a = X1(:,1);
        x1b = X1(:,2);
        x1c = X1(:,3);
        x1d = X1(:,4);
        %subplot(2,1,1);
        %plot(t,x1a,t,x1b,t,x1c,t,x1d)
        %xlabel('time');
        %ylabel('Perturbation value ');
        %legend('Sideslip (deg)','roll rate (deg/s)','bank angle (deg)','yaw rate
(deg/s)');
        %title('Lateral-directional Response to an Aileron Impulse Input');

        %figure;
        x2a = X2(:,1);
        x2b = X2(:,2);
        x2c = X2(:,3);
        x2d = X2(:,4);
        %subplot(2,1,2);
```

```
        %plot(t,x2a,t,x2b,t,x2c,t,x2d)
        %xlabel('time');
        %ylabel('Perturbation value ');
        %legend('Sideslip (deg)','roll rate (deg/s)','bank angle (deg)','yaw rate
(deg/s)');
        %title('Lateral-directional Response to an Aileron Step Input');

        % Calculate and plot response to rudder deflection
        % Model response to a step input
        [Y4,X4,t] = step(A,Br,Co,D,Dr,t);

        % Model response to a unit impulse
        [Y3,X3,t] = impulse(A,Br,Co,D,Dr,t);

        % Plot Dynamic Response to a Rudder Deflection
        %figure;
        x3a = X3(:,1);
        x3b = X3(:,2);
        x3c = X3(:,3);
        x3d = X3(:,4);
        %subplot(2,1,1);
        %plot(t,x3a,t,x3b,t,x3c,t,x3d)%normalized velocity
        %xlabel('time');
        %ylabel('Perturbation value ');
        %legend('Sideslip (deg)','roll rate (deg/s)','bank angle (deg)','yaw rate
(deg/s)');
        %title('Lateral-directional Response to a Rudder Impulse Input');

        %figure;
        x4a = X4(:,1);
        x4b = X4(:,2);
        x4c = X4(:,3);
        x4d = X4(:,4);
        %subplot(2,1,2);
        %plot(t,x4a,t,x4b,t,x4c,t,x4d)%normalized velocity
        %xlabel('time');
        %ylabel('Perturbation value ');
        %legend('Sideslip (deg)','roll rate (deg/s)','bank angle (deg)','yaw rate
(deg/s)');
        %title('Lateral-directional Response to a Rudder Step Input');
      end
    end

end

DRr1print = [real(DRollr1);imag(DRollr1)];
DRr2print = [real(DRollr2);imag(DRollr2)];
fprintf(fid1,'\n');
fprintf(fid1,'\n');
fprintf(fid1,'\n Lateral-Directional Dynamic Stability Results ');
fprintf(fid1,'\n ---------------------------------------');
fprintf(fid1,'\n Roll root                = %4.5f',Rollroot);
fprintf(fid1,'\n Roll time constant       = %4.5f',Troll);
fprintf(fid1,'\n Dutch Roll roots:    Real          Imaginery');
fprintf(fid1,'\n                      %4.5f       %4.5f',DRr1print);
fprintf(fid1,'\n                      %4.5f       %4.5f',DRr2print);
fprintf(fid1,'\n Dutch Roll damping ratio= %4.5f',DRdamp);
fprintf(fid1,'\n Dutch Roll natural freq.= %4.5f',DRnat);
fprintf(fid1,'\n Spiral root              = %4.5f %4.5f',Spiralroot);
```

C.25 Controls.m

This file calculates the control deflection requirements and is called by *analyze.m*.

```
%Controls.m
% This code determines the necessary control surface deflections under
% varying circumstances.  These deflections form constraints based
% on a user-defined maximum and minimum control surface travel.

% Elevator Control
DEmax = 25;
DEmin = -25;

% First concerned with static control
ff = [CLA CLDE; CMA CMDE];
gg = [cLmax-CL;-CM];
Deflection = inv(ff)*gg;
DEstatic = Deflection(2)*180/pi; % gives result in degrees

% Secondly concerned with takeoff rotation
Vrotate = 1.11*VTO;
Aground = atan((Ln/12-Lm/12)/(CGmgr-CGngr))+IWING*pi/180;
%Temp fix
CL0 = 0;
CM0 = -0.04; %Depends on Type of Airfoil
ACent = Xlewng+.3*CROOT;

DErotate = 180/pi*(Wguess*(CGmgr-XM)/(.5*densSL*Vrotate^2*S*MACwing)-CM0-CMA*Aground ...
       -CL0*(CGmgr-ACent)/MACwing-CLA*Aground*(CGmgr-ACent)/MACwing)/(CMDE + CLDE*(CGmgr-
ACent)/MACwing);

% Aileron Control
DAmax = 15;
DAmin = -15;

% Concerned with value of steady (static) roll rate from Schmidt
Pstatic = 5; %deg/sec

DAstatic = (-Pstatic*pi/180*LDA/LP);

% Rudder Control

% Concerned with control capability in a the crosswind condition
BetaMax = 10;
BetaMin = -10;
DRmax = 30;
DRmin = 30;

ff = [CL CYDR CYDA; 0 CLDR CLdeltaA; 0 CNDR CNDA];
gg1 = -[CYB; CLB; CNB]*BetaMax*pi/180;
gg2 = -[CYB; CLB; CNB]*BetaMin*pi/180;

Deflection1 = inv(ff)*gg1;
Deflection2 = inv(ff)*gg2;
DR1 = Deflection1(2)*180/pi;
DA1 = Deflection1(3)*180/pi;
DR2 = Deflection2(2)*180/pi;
DA2 = Deflection2(3)*180/pi;
```

C.26 Allconstraints.m

This file calculates and organizes all of the available constraints. This is called by *analyze.m*.

```
%Allconstraints.m
%This file assigns the full set of constraints to the g and geq vectors
CONSset = wk1read('UserConstraints.wk1');
SMmax = CONSset(1); SMmin = CONSset(2); CYBo = CONSset(3); CLBo = CONSset(4);
CNBo = CONSset(5); DEmax = CONSset(6); DEmin = CONSset(7); DAmax = CONSset(8);
DAmin = CONSset(9); DRmax = CONSset(10); DRmin = CONSset(11); PhDmin = CONSset(12);
SPmin = CONSset(13); SPmax = CONSset(14); SPnatmax = CONSset(15);
SPnatmin = CONSset(16);DRdmin = CONSset(17); Trollmax = CONSset(18);



g(1,1) = (YI+BA)/(.5*B) - 1;
g(2,1) = (YFlap+BFlap)/YI - 1;
g(3,1) = BT/B-1;
g(4,1) = BV/B-1;
g(5,1) = BE/(.5*BT)-1;
g(6,1) = BR/BV-1;
g(7,1) = CA/(.2*CROOT)-1;
g(8,1) = CHE/(.5*Crtail)-1;
g(9,1) = CR/(0.5*Crvt)-1;
g(10,1) = (abs(Xletail)+abs(Crtail))/abs(LB)-1;
g(11,1) = (abs(Xlevt)+abs(Crvt))/abs(LB)-1;
g(12,1) = (abs(Xlewng)+abs(CROOT))/abs(Xletail)-1;
g(13,1) = (abs(Xlewng)+abs(CROOT))/abs(Xlevt)-1;
g(14,1) = abs(XM)/abs(Xletail)-1;
g(15,1) = abs(CGngr/CGmgr) -1;
g(16,1) = abs(CGngr/XM) - 1;
g(17,1) = abs(XM/CGmgr) - 1;
g(18,1) = AE/6.5 - 1;
g(19,1) = -AE;
g(20,1) = ART/10 - 1;
g(21,1) = -ART;
g(22,1) = AR/8 - 1;
g(23,1) = 1-AR/4;
g(24,1) = StaticMargin/SMmax - 1;
g(25,1) = 1-StaticMargin/SMmin;
g(26,1) = CYB;
g(27,1) = CLB;
g(28,1) = -CNB;
g(29,1) = CYB/CYBo-1;
g(30,1) = CLB/CLBo-1;
g(31,1) = CNB/CNBo-1;
g(32,1) = abs(DEstatic)/DEmax-1;
g(33,1) = abs(DEstatic)/DEmin-1;
g(34,1) = -1 - DErotate/DEmin;
g(35,1) = DAstatic/DAmax-1;
g(36,1) = -1 - DAstatic/DAmin;
g(37,1) = DR1/DRmax - 1;
g(38,1) = -1-DR1/DRmin;
g(39,1) = DR2/DRmax-1;
g(40,1) = -1-DR2/DRmin;
g(41,1) = DA1/DAmax - 1;
g(42,1) = -1-DA1/DAmin;
```

```
g(43,1) = DA2/DAmax - 1;
g(44,1) = -1-DA2/DAmin;
g(45,1) = 1-Phugdamp/PhDmin;
g(46,1) = 1-SPdamp/SPmin;
g(47,1) = SPdamp/SPmax-1;
g(48,1) = SPnat/SPnatmax - 1;
g(49,1) = 1-SPnat/SPnatmin;
g(50,1) = 1-DRdamp/DRdmin;
g(51,1) = abs(Troll)/Trollmax-1;
g(52,1) = WmissFuel/Wfw - 1;
g(53,1) = -WmissFuel;

g = [g; gmission];

% Equality Constraints
geq(1,1) = tipangle/15 - 1;

geq(2,1) = clearance/15 - 1;

geq(3,1) = Pnose/Wguess-0.1;

geq(4,1) = Pmain/Wguess-0.9;

geq(5,1) = Wcalc/Wguess-1;

geq(6,1) = CGngr-abs(CGngr);
geq(7,1) = Lm - abs(Lm);
geq(8,1) = Ln - abs(Ln);
geq(9,1) = LB - abs(LB);
geq(10,1)= Xlewng - abs(Xlewng);
geq(11,1)= Xletail- abs(Xletail);
geq(12,1)= Xlevt - abs(Xlevt);


% Create output file for constraint values
fid3 = fopen('Output.txt','w');
fprintf(fid3,'\n Final Design Characteristics');
fprintf(fid3,'\n');
fprintf(fid3,'\n Static Margin                      = %4.6f',StaticMargin);
fprintf(fid3,'\n Static Elevator Deflection, deg    = %4.6f',DEstatic);
fprintf(fid3,'\n Rotation Elevator Deflection, deg  = %4.6f',DErotate);
fprintf(fid3,'\n Static Aileron Deflection, deg     = %4.6f',DAstatic);
fprintf(fid3,'\n Rudder Deflection (+beta), deg     = %4.6f',DR1);
fprintf(fid3,'\n Aileron Deflection (+beta), deg    = %4.6f',DA1);
fprintf(fid3,'\n Rudder Deflection (-beta), deg     = %4.6f',DR2);
fprintf(fid3,'\n Aileron Deflection (-beta), deg    = %4.6f',DA2);
fprintf(fid3,'\n Mission Fuel Burn, lb              = %4.6f',WmissFuel);
fprintf(fid3,'\n Tip Angle Criterion, deg           = %4.6f',tipangle);
fprintf(fid3,'\n Ground Clearance Criterion, deg    = %4.6f',clearance);
fprintf(fid3,'\n Nose Gear Loading, lb              = %4.6f',Pnose);
fprintf(fid3,'\n Nose Gear Loading Ratio            = %4.6f',Pnose/Wguess);
fprintf(fid3,'\n Main Gear Loading, lb              = %4.6f',Pmain);
fprintf(fid3,'\n Main Gear Loading Ratio            = %4.6f',Pmain/Wguess);
fprintf(fid3,'\n Calculated Total Weight, lb        = %4.6f',Wcalc);
fprintf(fid3,'\n Approximated Empty Weight, lb      = %4.6f',We2);
fprintf(fid3,'\n Calculated Empty Weight, lb        = %4.6f',We1);
fprintf(fid3,'\n');
fprintf(fid3,'\n');
fprintf(fid3,'\n Results of Raymer Component Weight Estimation');
fprintf(fid3,'\n');
fprintf(fid3,'\n Total Aircraft Weight, lb   = %4.2f',Wtotal);
fprintf(fid3,'\n Wing Weight, lb             = %4.2f',Wwing);
fprintf(fid3,'\n Horizontal Tail Weight, lb  = %4.2f',Wht);
```

```
fprintf(fid3,'\n Vertical Tail Weight, lb    = %4.2f',Wvt);
fprintf(fid3,'\n Fuselage Weight, lb         = %4.2f',Wfuse);
fprintf(fid3,'\n Main Gear Weight, lb        = %4.2f',Wmain);
fprintf(fid3,'\n Nose Gear Weight, lb        = %4.2f',Wnose);
fprintf(fid3,'\n Engine Weight, lb           = %4.2f',Wengine);
fprintf(fid3,'\n Fuel System Weight, lb      = %4.2f',Wfuel);
fprintf(fid3,'\n Flight Controls Weight, lb  = %4.2f',Wcontrol);
fprintf(fid3,'\n Hydraulics Weight, lb       = %4.2f',Whydra);
fprintf(fid3,'\n Avionics Weight, lb         = %4.2f',Wavion);
fprintf(fid3,'\n Electrical System Weight,lb = %4.2f',Welec);
fprintf(fid3,'\n AC and Anti-ice Weight, lb  = %4.2f',Wac);
fprintf(fid3,'\n Furnishings weight, lb      = %4.2f',Wfurn);
fprintf(fid3,'\n');
fprintf(fid3,'\n Calculated CG position, ft  = %4.4f',Xcg);
fprintf(fid3,'\n');
fprintf(fid3,'\n Constraint Vector Values');
count = 1:1:length(g);
count = transpose(count);
count1 = transpose([count g]);
fprintf(fid3,'\n %2.0f   %4.4f',count1);
fprintf(fid3,'\n');
count = 1:1:length(geq);
count = transpose(count);
count2 = transpose([count geq]);
fprintf(fid3,'\n %2.0f   %4.4f',count2);
fprintf(fid3,'\n');
fprintf(fid3,'\n Design Variable Values');
count = 1:1:length(DV);
count = transpose(count);
count3 = transpose([count DV]);
fprintf(fid3,'\n %2.0f   %4.4f',count3);
```

C.27 plotdesign.m

This file was developed to allow the user to obtain a picture of the resulting design. It is a work in progress and has not yet been linked to the overall program. It's use is outlined in Appendix B.

```
% plotdesign.m
% graphically depict the resulting designs

% Calculate tip chords
Ctipwing = TR*CROOT;
Ctiptail = TRT*Crtail;
Ctipvert = TRvt*Crvt;

% Convert sweeps to radians
SA = PP(27);SAH = PP(30);SAvt = PP(32);

SA = SA*pi/180;
SAH = SAH*pi/180;
SAvt = SAvt*pi/180;

% Draw the fuselage
```

```
Xfuselage = [0 2 LFCY LBCY LMH .9*LB LB LB];
Yfuselage = [0 WNOSE/2 WFCY/2 WFUS/2 WBCY/2 R1 R1/3 0];
patch(Xfuselage, Yfuselage,'b');
patch(Xfuselage, -Yfuselage,'b');
axis([-2,30,-20,20]);
axis equal

% Draw the wing
xa = Xlewng;
xb = Xlewng+tan(SA)*B/2;
xc = xb+Ctipwing;
xd = Xlewng+CROOT;

ya = 0;
yb = (B/2)/cos(SA);
yc = yb;
yd = 0;

Xwing = [xa xb xc xd];
Ywing = [ya yb yc yd];
patch(Xwing,Ywing,'r');
patch(Xwing,-Ywing,'r');

% Draw the flaps
beta = atan(yb/(xd-xc));
Xflap = [xd-CFlap-cos(beta)*YFlap xd-CFlap-cos(beta)*(BFlap+YFlap) ...
        xd-(BFlap+YFlap)*cos(beta) xd-cos(beta)*YFlap];
Yflap = [YFlap YFlap+BFlap*sin(beta) YFlap+BFlap*sin(beta) YFlap];
patch(Xflap,Yflap,'g');
patch(Xflap,-Yflap,'g');

% Draw the ailerons
yy = YFlap+BFlap*sin(beta)+.2;
Xail = [xd-CA-cos(beta)*YI xd-CA-cos(beta)*(BA+YI) xd-(BA+YI)*cos(beta) xd-cos(beta)*YI];
Yail = [yy yy+BA*sin(beta) yy+BA*sin(beta) yy];
patch(Xail,Yail,'g');
patch(Xail,-Yail,'g');

% Draw the horizontal tail
xa = Xletail;
xb = Xletail+tan(SAH)*BT/2;
xc = xb+Ctiptail;
xd = Xletail+Crtail;

ya = 0;
yb = (BT/2)/cos(SAH);
yc = yb;
yd = 0;

Xtail = [xa xb xc xd];
Ytail = [ya yb yc yd];
patch(Xtail,Ytail,'r');
patch(Xtail,-Ytail,'r');

% Draw the elevator
% Draw the ailerons
beta = atan(yb/(xd-xc));
Xail = [xd-CHE xd-CHE-cos(beta)*(BE) xd-(BE)*cos(beta) xd];
Yail = [0 BE*sin(beta) BE*sin(beta) 0];
patch(Xail,Yail,'g');
patch(Xail,-Yail,'g');
```

```
%Draw the prop
Xprop = [0 0 .1 .1];
Yprop = [0 Dprop/2 Dprop/2 0];
%patch(Xprop,Yprop,'k');
%patch(Xprop,-Yprop,'k');

%% BEGIN SIDE VIEW DRAWING
figure;
Xfuse1 = [1,0, 1, .5*LB,LB];
Yfuse1 = [0,-1.5,-3,-3,0];
patch(Xfuse1, Yfuse1,'b');

axis([-2,30,-8,8]);
axis equal
SAvt = 50*pi/180;
%Draw veritical tail
xa = Xlevt;
xb = Xlevt+tan(SAvt)*BV/2;
xc = xb+Ctipvert;
xd = Xlevt+Crvt;

ya = 0;
yb = (BV)*cos(SAvt);
yc = yb;
yd = 0;

Xvert = [xa xb xc xd];
Yvert = [ya yb yc yd];
patch(Xvert,Yvert,'r');

%Draw the rudder
% Draw the elevator
% Draw the ailerons
beta = atan(yb/(xd-xc));
Xrud = [xd-CR xd-CR+abs(cos(beta))*(BV) xd+(BV)*abs(cos(beta)) xd];
Yrud = [0 yb yb 0];
patch(Xrud,Yrud,'g');

%Draw the wing
Xwing = [Xlewng Xlewng Xlewng+.25*CROOT Xlewng+CROOT Xlewng+CROOT];
Ywing = [0 .2 .4 .1 0];
patch(Xwing,Ywing,'y');

%Draw the horiz. tail
Xtail = [Xletail Xletail Xletail+.25*Crtail Xletail+Crtail Xletail+Crtail];
Ytail = [0 .2 .4 .1 0];
patch(Xtail,Ytail,'y');

%Draw landing gear
Xngr = [CGngr CGngr CGngr+.2 CGngr+.2];
Yngr = [-3 -3-Ln/12 -3-Ln/12 -3];
patch(Xngr,Yngr,'k');

Xmgr = [CGmgr CGmgr CGmgr+.2 CGmgr+.2];
Ymgr = [-3 -3-Lm/12 -3-Lm/12 -3];
patch(Xmgr,Ymgr,'k');
```